Evaluation of KPI Monitoring Tools for an Automotive Supplier

Master Thesis

submitted in fulfilment of the requirements for the academic degree

M.Sc. in Automotive Software Engineering

Department of Computer Science

Professorship of Computer Engineering

Submitted by:	Hasnain Haider Baig, Mirza
Matrikel-Nr:	303912
Professor:	Prof. Dr. W. Hardt
Supervisors:	Dr. Ariane Heller
	Dipl. Wirt.-Inf. Uwe Kirst,
	Mentor Graphics Development (Deutschland) GmbH

Declaration of Authorship

I, Hasnain Haider Baig Mirza, assure that the thesis “Evaluation of KPI Monitoring Tools for an Automotive Supplier” is my own work under the guidance of my supervisors. The data collected during the literature review and referred in this document is given due acknowledgement. All the references and helping materials are enlisted in the Bibliography with all sincerity.

Signature: _____

Date: _____

Abstract

Automotive SPICE is used to evaluate the efficiency of the development processes for OEMs and ECU suppliers. It is a domain specific version of SPICE. Many automotive manufacturers are demanding Automotive SPICE level 2 from their suppliers. Application life cycle management solutions specific to automotive industry focus more on standardisation yet providing very little information about performance of processes. This thesis shows how Business Intelligence (BI) solutions can be helpful in achieving Automotive SPICE level 2 compliance and as comprehensive reporting tools. Moreover the evaluation of approaches for implementation of different BI systems has shown that commercial solutions can be beneficial for small to medium size suppliers. Implementing a BI solution can also point out possible improvement opportunities and bad practices for an organisation. The implemented BI system can only be efficient and useful if the underlying data is accurate and responsive.

Acknowledgements

In the name of Allah, the Most Gracious, the Most Merciful.

First I would like to give my deep gratitude to the people around me during the work of my master thesis. Secondly, I would like to show my appreciation to my mum, who showed her encouraging attitude during the entire span of my master program with her affections and moral support. I am and will be grateful to my dad (who passed away), for his love and support. I cannot thank enough to my supervisor Mr. Uwe Kirst at Mentor Graphics Automotive Business Unit in Villingen-Schwenningen, who was abundantly helpful and offered invaluable guidance throughout my work. Indeed something as complex as Master thesis could not have been completed without his suggestions, comments, dedicated time and his always responsive behaviour. I would also like to take this opportunity to express my gratitude to Mr. Michael Maier for granting me extra time for the completion of my thesis due to my health issues. Moving to the academic side, my Professor, Dr. Wolfram Hardt contributed to the development of my skills even before my thesis had started. These skills certainly helped me better understand the engineering problems and taught me the art of technical documentations. Furthermore, Dr. Ariane Heller was great source of motivation especially because of the points she raised during my concept presentation and her guidance regarding the official matters which helped a lot in the management of thesis.

Table of Contents

List of Figures

List of Tables

Abbreviations

Full Form	Abbreviation
Accessing Organization	AO
Application Lifecycle Management	ALM
Application Programming Interface	API
Auto motive Special Interest Group	AUTOSIG
Business Intelligence	BI
Capability Maturity Model Integration	CMMI
Data Analysis Expressions	DAX
Electronic Control Unit	ECU
Enterprise Resource Planning	ERP
eXtensible Markup Language	XML
Extract Transform Load	ETL
Field-programmable gate array	FPGA
Generic Practice	GP
Herstellerinitiative Software	HIS
International Electrotechnical Commission	(IEC)
International Standards Organization	ISO
JavaScript Object Notation	JSON
Key Performance Indicator	KPI
Massachusetts Institute of Technology	MIT
MS	Microsoft
OnLine Analytic Processing functionality	OLAP
Original Equipment Manufacturer	OEM
Performance Measurement System	PMS
Process Assessment Model	PAM
Process Attribute	PA
Process Reference Model	PRM
Project Management Professional	PMP
Query Language	SQL
REpresentational State Transfer	REST
Software as a Service	SaaS
Software Development Kit	SDK
Software Process Improvement and Capability dEtermination	SPICE
SQL Server Reporting Services	SSRS
Static Code Analysis	SCA
Structured Query Language	SQL
Technical Requirement Specification	TRS
Unified Modeling Language	UML

Verband der Automobilindustrie	VDA

Table 1.0.1: Abbreviations

Chapter 1

1. Introduction

"You can't control what you can't measure"[1]

Nowadays software have become vital part of daily routines. We trust software with our lives in safety critical areas like hospitals, aerospace and auto mobiles. Developing any software for safety critical systems is a delicate task. Software development companies are facing tough challenges like high initial investment, budget shortages, cancellation of projects and most importantly the quality of the product. In automotive industry if final products are delivered with faulty software, it can lead to high compensation costs or in worst case scenario causalities.

The quality of software product is directly related to quality of its development process. The major concern of automotive manufacturers is to not only improve quality of the software but also their software process management. It's unlikely for an automotive manufacturer to complete whole product on its own. The final product may contain software from many other Original Equipment Manufacturers (OEMs) and suppliers. The reason behind dividing work load is to reduce the cost, increase productivity and take benefit from expertise in certain areas e.g. in some vehicles Volkswagen is using fuel injectors from Bosch. This approach comes with overheads like management of commutation, communication interfaces, coordination, not meeting deadlines and lacking desired quality. These problems could be reduced by introducing a process to implement and control engineering, quality assurance, acquisition and cooperation with external partners. [1]

With recent breakthroughs, automotive software industry is saturating really fast in terms of innovation. To stay on top, automotive manufacturers are bound to continuously improve their software process. Software process improvement involves realizing either right process is being followed and then evaluation of issues in that process. Software process improvement is a cyclic procedure and there is always need to know that what and how it can be improved.

The modern age of software development is coming up with lot of software development lifecycles. The two main categories of these methodologies are traditional and agile. It is not essential for an organisation to adapt these methodologies completely, as organisation can adopt, tailor and improve these processes accordingly.

[1] Tom DeMarco, Controlling Software Projects: Management Measurement & Estimation, (1982), p. 3.

International Standards Organization/ International Electrotechnical Commission (ISO/IEC) 15504 also acknowledged as Software Process Improvement and Capability dEtermination (SPICE) and its derived version known as Automotive SPICE are software process assessment models mostly used in Europe among automotive manufacturers, OEMs or tier N suppliers. On the other hand there is Capability Maturity Model Integration (CMMI) but details and comparison of Automotive SPICE with CMMI is outside the scope of this thesis.

This thesis discusses importance of Automotive SPICE compliance. It also discusses evaluation and implementation of tools used for automated Key Performance Indicators (KPIs) collection to help in achieving Automotive SPICE level 2. In Section (5.2), an example project from major tier 1 supplier for an OEM is taken into account during research work in the development environment of an automotive supplier.

1.1 Motivation

The purpose behind initiation of this topic is to increase productivity and improve service provided by automotive supppliers. Another important reason is to satisfy concerns of customers in more managed fashion. Many projects from different customers require certain level of Automotive SPICE compliance.

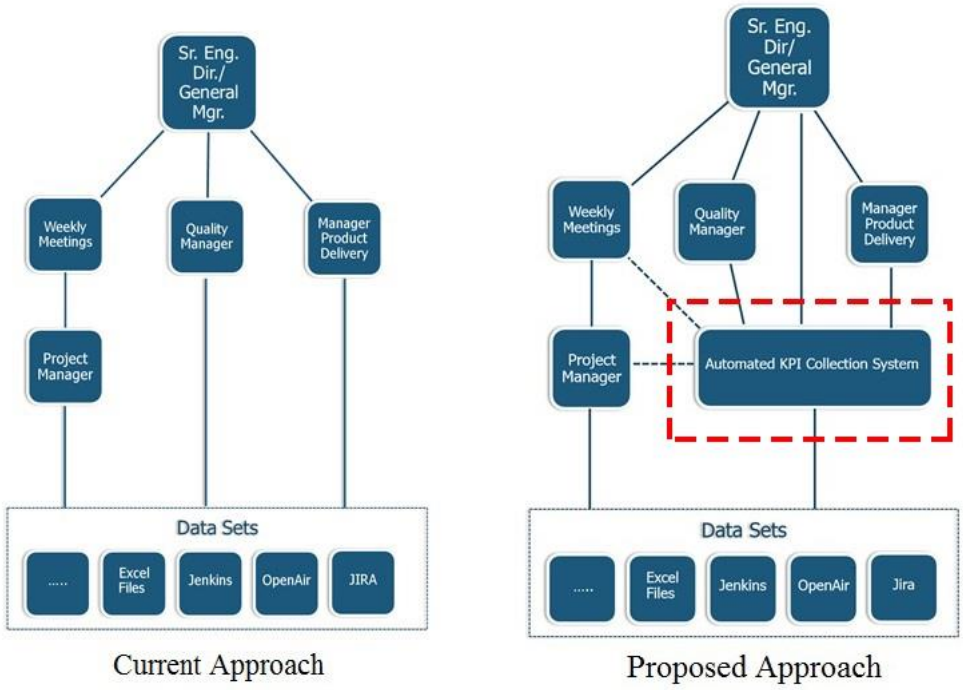


Figure 1.1: Current and proposed approaches

Defining KPIs in a project is responsibility of quality assurance plus process management team, but there is still need for technical areas to be handled in software process improvement.

As shown in Figure 1.1 the current structure of the organisation lacks reporting and transparency to higher management. The higher management relies solely on bi-weekly reports or personal meetings with project managers. The performance of process is also not available.

Problems faced by an organisation missing automated KPIs collection system are following:

- With increasing number of projects manual reporting is becoming more and more difficult and time consuming
- Real time status of projects is not available
- The reports from project management tools are not comprehensive and customizable
- SPICE level 2 compliance is difficult to achieve
- Higher level management have no sufficient information for quick decision making

The Figure 1.1 also shows the proposed approach to solve the problem. A tool is introduced which can automatically collect data for KPIs and report it to the higher management in a graphical fashion. This approach can be beneficial in following ways:

- Automated collection of KPIs which can lead to SPICE level 2 compliance
- Scheduled updates on KPIs
- Help creating more compressive reports
- Helping higher management making fast and meaningful decisions

1.2 Problem Statement

The research work carried out during this thesis is to help organisations in achieving SPICE level 2. This doesn't mean SPICE level 2 compliance could completely be acquired using automated KPIs collection tool. SPICE level 2 compliance is discussed in details in Chapter 2.

The problem this thesis addresses requires both management and technical skills. It is also intended to come up with a solution which not only helps in achieving SPICE level 2 but also helps organisation in its normal operations.

In current decade Business Intelligence (BI) tools gained a lot of popularity because of exponential growth in information technology sector. BI tools are good way to grab and visualize important KPIs. It's been realized that developing an in-house solution for collection of important business information requires great amount of effort. It is also not cost effective as well as involves intensive resources for maintenance. Developing a solution from scratch is not only outside the time limit of this thesis it also requires extra resources. Developing or adopting the KPIs collection solution is trade-off between resources required to build or buy.

It is also neither intended nor possible to adopt out of the box solution without spending any effort on tailoring it according to specified requirements. The modern version of BI tools provide integration with other project management tools and web services e.g. Microsoft (MS) project manager, share point, Google analytics, etc. Most of these tools provide Application Programming Interfaces (APIs) for communicating with external sources and services.

Data integration is most important thing when it comes to the implementation of a BI tool. Usually data is scattered within organisations and also among external partners and OEMs. Integration of data in a way that it can be interpreted into something meaningful and without compromising its security, is an enormous challenge. The major reason behind failure of most BI projects is failing to integrate data from different resources. There is an option available between integrating data using data warehouses or data marts, establishing database server or using API calling.

The problem statement can be categorized into following three main challenges:

- Gathering requirement to understand current environment and approaches within organisation and proposing the suitable approach
- Evaluation of BI tools as recently there are several famous and hundreds of small BI solutions available. The evaluation for a BI tool involves effort required for its implementation, available support, requirements satisfaction and maintenance cost
- Implementation of BI tool on in house projects and assessment of level of SPICE compliance

The quality of product is really vital in automotive industry and it is depends on the underlying process. Software process improvement is a continuous process, so it's tough to say that an organisation had achieved ideal or perfect development process.

Therefore, with the above mentioned problems, this thesis investigates the following questions:

- Can BI tool help in achievement of Automotive SPICE capability Level 2?
- Commercial or open source solution?
- Which approach is more efficient for an automotive supplier i.e. either data collection through data marts or API calling to gather data from data sets?
- Impact of solution on current processes?

Automotive SPICE Capability levels are explained in details in Section (2.1.2).

1.3 Contribution of Thesis

This thesis mainly addresses two parts of automotive software development process, Automotive SPICE compliance and business intelligence. It's been proposed that to achieve level 2 SPICE business intelligence systems are most suitable way to gather KPIs. The key contributions of this thesis are summarized as follows:

- Using BI tools to help achieving SPICE level 2
- Increasing transparency for higher level management to the performance of software development processes and status of different KPIs
- Evaluation of major BI tools and data integration/collection techniques comparison

Data integration plays major role in success of a BI project implementation. Thanks to APIs, JavaScript Object Notation (JSON) or eXtensible Markup Language (XML) data can be collected from different resources and services, which was only possible before using data marts and local database servers. Establishing a dedicated database server is more concrete way to data integration but it can cause extra resources, high maintenance cost and with risk of it becoming obsolete too fast.

With technology changing automotive suppliers are decreasing their product roll out time. This is putting quality and safety under more stress. In recent years major automotive manufacturers have faced large penalties in terms of financial setbacks and losing reputation due to lack of transparency of software development processes. Thus process compliances like Automotive SPICE are now important more than ever.

1.4 Structure of Thesis

The rest of thesis is structure as follows:

Chapter 2 describes introduction and importance of SPICE, Automotive SPICE and Automotive SPICE compliance. Furthermore it contains concept of KPIs and their importance in automotive software development units. Additionally datasets are described with introduction to famous BI tools.

Chapter 3 explains state of the art current techniques in software development process by automotive suppliers at SPICE level 1. It also addresses issues with current environment and side effects because of those issues.

Chapter 4 states problems and elaborates them in the light of functional and non-functional requirements. Addressing major challenges faced by implementation of BI tool for KPIs collection in current environment.

Chapter 5 explains architecture and methodology of KPIs collection system. Furthermore an example project is taken into account while explaining methodology in details.

Chapter 6 explains actual work done during the research work, considering the development environment of Mentor Graphics automotive business unit situated in Germany. Furthermore evaluation of BI tools and data integration techniques are explained in details.

Chapter 7 and 8 outlines results, assessment and effects of BI tool's implementation. Furthermore API calling and future roadmap is described.

Chapter 2

2. Background

In nineties the massive surge in usage of software urged International Standards Organisation (ISO) to form a committee in 1993 to establish standards for software process improvement and capability measurement. This committee proposed set of standards known as ISO/IEC 15504 also known as SPICE. SPICE initially meant Software Process Improvement and Capability Evaluation, because of meaning of evaluation in French the new abbreviation is interpreted as Software Process Improvement and Capability dEtermination. In coming years different industries adopted SPICE according to their domain specific needs. The noticeable variants include Automotive SPICE, SPICE 4 Space, and SPICE in medical systems. [2]

ISO/IEC 15504 is the reference for assessors to determine the capability of organisation for delivering services, software or systems. The practices adopted by organisation are compared with reference mutuality models provided by SPICE.[2] The reference models vary from domain to domain.

Automotive SPICE has its own Process Reference Model (PRM) and Process Assessment Model (PAM). [3] This chapter briefs about Automotive SPICE importance, Automotive SPICE levels of compliance, performance management, introduction to Key Performance Indicators (KPIs), overview of Business Intelligence (BI) tools and data sets used during the thesis.

2.1 Automotive SPICE

Nowadays modern automobiles have ever increasing demand of quality, performance, latest and innovative technology and economical worth. With increased competition automotive suppliers are under constant pressure of rolling out new models in short time periods. Still functionality cannot be sacrificed because of increased cost and complexity. [4]

Automotive manufacturers soon realized that they can determine capability of their vendors by determining the quality of the process. By evaluating adopted processes of vendors, manufacturers can improve the contractual clauses and assess feasibility of goods or services promised to be provided in given deadline.

[2] ISO/IEC 15504-2 Clause 5

Some automotive software standards like AUTomotive Open System ARchitecture (AUTOSAR) are targeted to standardise operating system architecture in automotive software industry and improve reusability and quality of product. Some other noticeable standards are:

- "Offene Systeme und deren Schnittstellen für die Elektronik in Kraftfahrzeugen (OSEK)"
- CMMI
- ISO-26262

Looking at increased use of software in automobiles Automotive SPICE was developed in 2001 through the AUTOSIG (Auto motive Special Interest Group) by automobile manufacturer Daimler, Audi, BMW, Porsche, Volkswagen, Fiat , Ford , Jaguar , land Rover, Volvo and the SPICE User Group and the procurement forum.

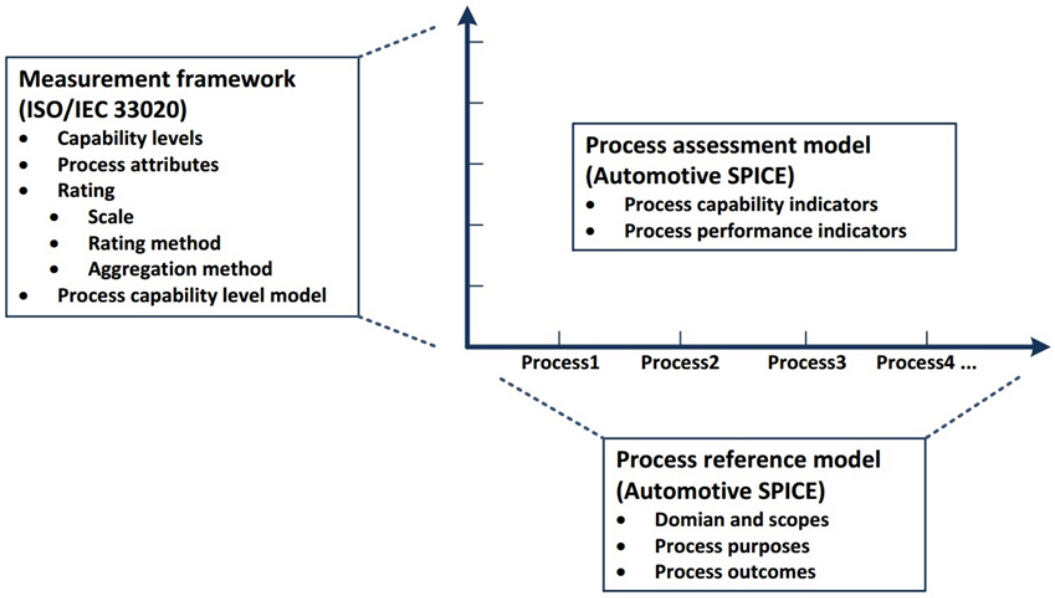


Figure 2.1: SPICE Process Assessment Model [5, Page 11]

Automotive SPICE is a domain-specific version of the SPICE. The purpose of Automotive SPICE is to evaluate the efficiency of the development processes for Electronic Control Unit (ECU) suppliers in the automotive business. "Verband der Automobilindustrie (VDA)" is the owner of Automotive SPICE in Europe, USA and Switzerland. Other than SPICE, CMMI has also become progressively widespread in the automobile production.

As shown in Figure 2.1, the concept of process capability determination by using a process assessment model is based on a two-dimensional framework. The first dimension is provided by processes defined in a process reference model (process dimension). The second dimension consists of capability levels that are further subdivided into process attributes (capability dimension). The process attributes provide the measurable characteristics of process capability. [5]

2.1.1 The Process Dimension

Processes are grouped by process category and at a second level into process groups according to the type of activity they address. There are 3 process categories: Primary Life Cycle Processes, Organisational Life Cycle Processes and Supporting Life Cycle Processes. [5] Figure 2.2 shows the process reference model provided by Automotive SPICE.

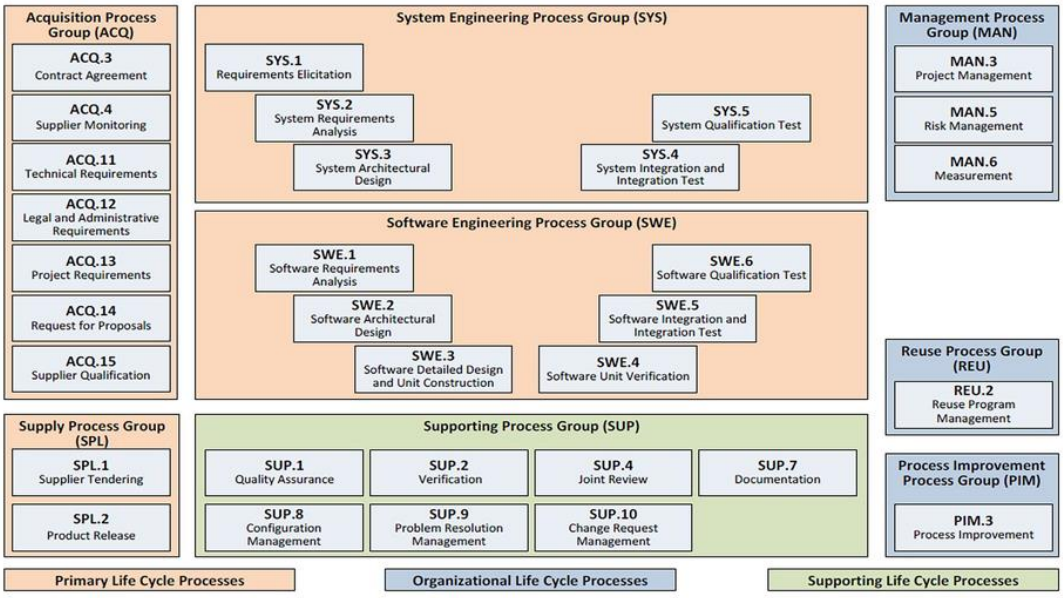


Figure 2.2: Automotive SPICE process reference model overview [5, Page 12]

Automotive SPICE also specifies structure and attributes of process e.g. process purpose, process outcomes, base practices and output work products. The German automobile manufacturers Audi, BMW, Daimler, Porsche and Volkswagen who are member of the "Herstellerinitiative Software (HIS)" have a common settlement of lowest level of assessment to be considered in processes. Since 1 January 2007 it had been accepted by the members of HIS to use the SPICE for the evaluation of suppliers in the software and electronics sector. One of the major advantages of SPICE is that mutually agreed models can be developed targeted to respective domain. Figure 2.3 shows the mapping of processes from SPICE to Automotive SPICE as required by HIS group.

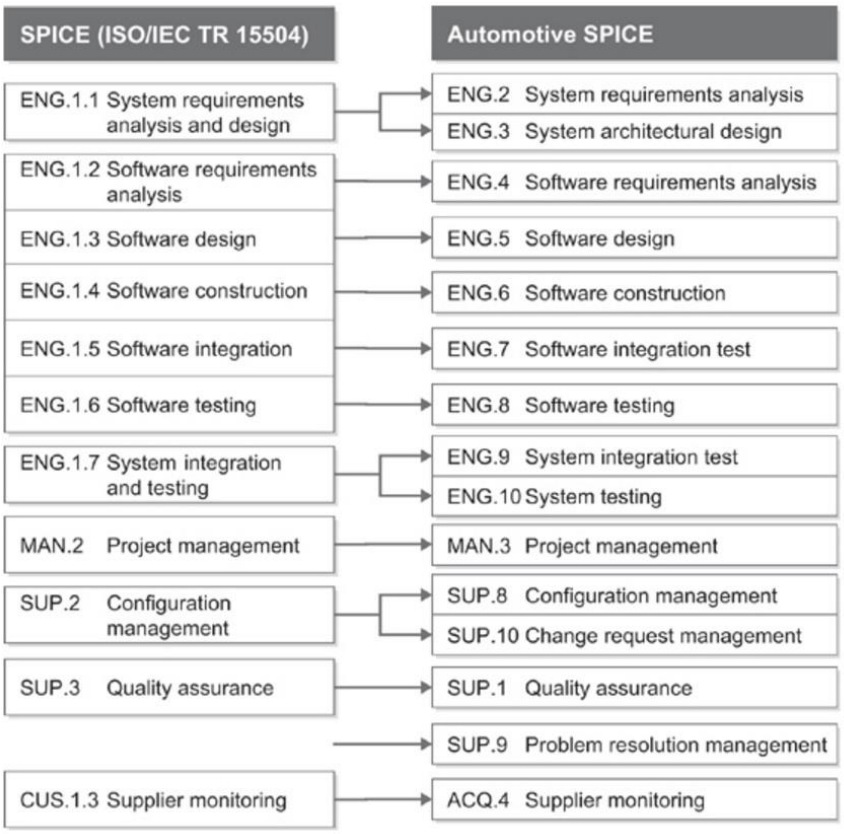


Figure 2.3: Mapping SPICE to Automotive SPICE [1, Figure 1-2]

"A possible trend can be recognized: the expansion from the successful application in software development to further domains, in particular to system development, mechanics, and hardware development. It stands to reason to apply in other development areas what has been tried and tested in the software domain. Whereas SPICE is currently preparing to expand towards system development with its new part 6, CMMI has already completed this move (software, hardware, systems, and services). Many enterprises, especially in Asia, did this with CMMI, thereby demonstrating their ability to develop high quality software efficiently. Now this trend is also beginning with Automotive SPICE. One example is Continental Automotive Singapore, Pte Ltd who recently achieved Automotive SPICE Level 4 in several processes". [1, Section 1.2]

The automotive industry is gradually shifting towards Automotive SPICE from SPICE. CMMI will remain its biggest competitor and adaptation of Automotive SPICE or CMMI solely depends upon business strategies of companies. This thesis does not discuss detailed comparison of CMMI and Automotive SPICE for further comparison studies refer to [6].

	Organization	Favored Process
BMW	E/E Process Chain	CMMI in addition to specific BMW targets
Chrysler	E/E Core Engineering	CMMI/SPICE
General Motors	Powertrain, Vehicle Engineering	GM-specific, based on parts of CMMI
Honda	E/E Systems R&D	Considering CMMI and other technologies
Mercedes Car Group	USA Germany	CMMI SPICE
Toyota		Proprietary, based somewhat on CMMI
Volkswagen	E/E Engineering R&D	SPICE
Bosch Automotive	All software business units, plus component development departments	CMMI/SPICE
Continental Automotive Systems	All business units that deliver software	SPICE
Delphi	Electronics & Safety	CMMI
Siemens VDO	All13 divisions	CMMI
Valeo	All devisions that deliver software	CMMI/SPICE
Visteon		Visteon Engineering Process (VEP), which uses elements of CMMI and SPICE

Table 2.1: Distribution of maturity models in the automotive industry [7]

2.1.2 The Capability Dimension

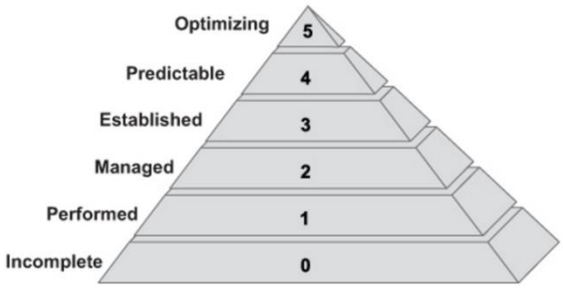


Figure 2.4: Capability levels [1, Figure 1-6]

Automotive SPICE provides six capability levels for assessment of processes. Each capability level has different process attributes. The success of compliance for any capability level depends upon success of these process attributes. Different process attributes are responsible for different feature of a capability level.

The official definitions of capability levels as described in Automotive SPICE Process Reference Model are described in Table 2.2.

Level 0: Incomplete process	The process is not implemented, or fails to achieve its process purpose.
Level 1: Performed process	The implemented process achieves its process purpose
Level 2: Managed process	The previously described performed process is now implemented in a managed fashion (planned, monitored and adjusted) and its work products are appropriately established, controlled and maintained.
Level 3: Established process	The previously described managed process is now implemented using a defined process that is capable of achieving its process outcomes.
Level 4: Predictable process	The previously described established process now operates predictively within defined limits to achieve its process outcomes. Quantitative management needs are identified, measurement data are collected and analysed to identify assignable causes of variation. Corrective action is taken to address assignable causes of variation.
Level 5: Innovating process	The previously described predictable process is now continually improved to respond to organisational change.

Table 2.2: Process Capability Levels [5, Page 15-16]

The amount of achievement for process attributes is measured by four scale points as given in ISO/IEC 33020. These points as defined in Automotive SPICE Process Reference Model are presented in Table 2.3.

Sometimes partially and largely achieved are further distinguished into (P-), (P+), (L+) and (L-). Automotive SPICE uses ISO/IEC 33020 as compliant Measurement Framework. As assessment of processes is not part of this thesis so different method provided in Automotive SPICE process reference model won't be discussed in detail here.

N	Not achieved	There is little or no evidence of achievement of the defined process attribute in the assessed process	0 to ≤ 15% achievement
P	Partially achieved	There is some evidence of an approach to and some achievement of, the defined process attribute in the assessed process. Some aspects of achievement of the process attribute may be unpredictable	> 15% to ≤ 50% achievement
L	Largely achieved	There is evidence of a systematic approach to and significant	> 50% to ≤ 85%

		achievement of, the defined process attribute in the assessed process. Some weaknesses related to this process attribute may exist in the assessed process	achievement
F	Fully achieved	There is evidence of a complete and systematic approach to, and full achievement of, the defined process attribute in the assessed process. No significant weaknesses related to this process attribute exist in the assessed process	> 85% to ≤ 100% achievement

Table 2.3: Rating scale according to ISO/IEC 33020 [5, Page 17]

2.2 Achieving Level 2 Automotive SPICE Compliance

Automotive industry has seen one of the fastest growths in software innovation. From E-Mobility to autonomous driving and from software controlled safety critical systems to state of the art infotainment systems, more or less 85% of the functionality in the modern motor vehicle now controlled by software, both the motor vehicle manufacturer and the supplier need to take action to address the concerns of potential recall and safety issues due to electronic systems malfunction. [4]

The scenario described here is based on personal experience gained during working on this thesis and may vary accordingly in different companies. In a supplier company at the start of a project, project leader with help of quality/process improvement manager selects set of processes for the project. These processes depend on type of the project. Next according to mutually agreed level of Automotive SPICE compliance process attributes are selected.

The goal of this thesis is to help achieving SPICE level 2 compliance by attaining performance management for processes in use. This thesis discusses process attributes up to level 2 only. Before discussing process attributes in details we need to understand concept of customer, basic practices and work product. The term customer has different meanings in different contexts. For manufacturers customer means end user who buys and uses the product. As this thesis is carried out from supplier's perspective, the term customer as mentioned otherwise refers to OEM or different department who are recipient for outcome of the product.

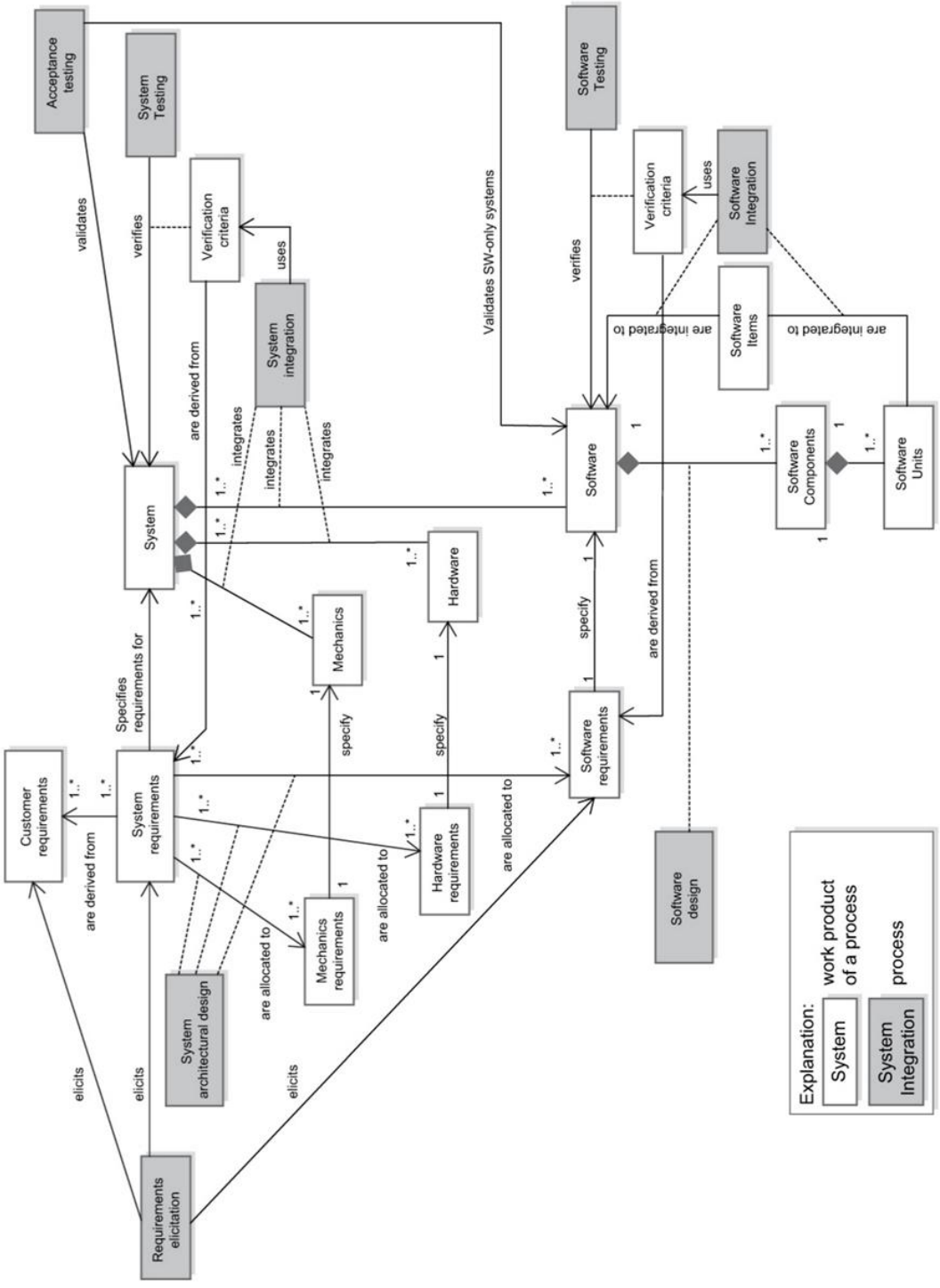


Figure 2.5: Example of engineering processes in Automotive SPICE [1, Figure 2-2]

Work products and base practices are also known as process performance indicators. They are process specific and not generic. Examples of base practices are reviewing contract, approval of contract or specifying rights and duties. Work product is different from deliverables.

A project may have many deliverable which could be handed over to stakeholders for further testing while work product is like an analysis and can be presented to stakeholders in the form of presentation, discussion, diagram or prototype. Work product need approval before its delivery. Some simple work products can be test plan, source code, weekly reports etc.

Moreover there are generic practices and generic resources associated with process attributes know as process capability indicators. These indicator help assessors in the process of process attribute rating and can be applied to any process. Generic practices are guidance for implementing a process attribute where generic resources are means which can be used during generic practices like tools, human resources or infrastructure.

Figure 2.5 elaborates a real life scenario of engineering processes as conceptualized by Automotive SPICE. A single process can deliver multiple work products or several processes can deliver single work product. Work products can be derived from existing work products.

At level 1 there is only one process attribute called Process PA(Process Attribute) 1.1 (process performance attribute). PA 1.1 process attribute should be full or largely achieved for every process to achieve level 1 compliance successfully.

PA 1.1 as described in [5, Page 78] as follows:

PA 1.1: Process performance, process attribute

The process performance attribute is a measure of the extent to which the process purpose is achieved. As a result of full achievement of this attribute:

a) the process achieves its defined outcomes

Generic practices	**GP(Generic Practices) 1.1.1 achieve the process outcomes** [ACHIEVEMENT a] Achieve the intent of the base practices Produce work products that evidence the process outcomes
Generic resources	**Resources are used to achieve the intent of process specific base practices** [ACHIEVEMENT a]

Table 2.4: PA 1.1 generic practices and resources [5, Page 78]

At level 2, PA 1.1 should be fully with newly introduced attributes PA 2.1 and PA 2.2 should be largely achieved. Level 1 focuses more on work products where level 2 motivates management of process performance and work products. Another major part of level 2 is tracking and monitoring of process performance and work products.

PA 2.1 is described in [5, Page 79] as follows:

PA 2.1 Performance management, process attribute

The performance management process attribute is a measure of the extent to which the performance of the process is managed. As a result of full achievement of this process attribute:

a) Objectives for the performance of the process are identified

b) Performance of the process is planned

c) Performance of the process is monitored

d) Performance of the process is adjusted to meet plans

e) Responsibilities and authorities for performing the process are defined, assigned and communicated

f) Personnel performing the process are prepared for executing their responsibilities

g) Resources and information necessary for performing the process are identified, made available, allocated and used

h) Interfaces between the involved parties are managed to ensure both effective communication and clear assignment of responsibility

Generic practices	**GP 2.1.1 Identify the objectives for the performance of the process.** [ACHIEVEMENT a] Performance objectives are identified based on process requirements. The scope of the process performance is defined. Assumptions and constraints are considered when identifying the performance objectives. NOTE 1: Performance objectives may include (1) timely production of artifacts meeting the defined quality criteria, (2) process cycle time or frequency (3) resource usage; and (4) boundaries of the process. NOTE 2: At minimum, project performance objectives for resources, effort and schedule should be stated.

	GP 2.1.2 Plan the performance of the process to fulfil the identified objectives. [ACHIEVEMENT b] Plan(s) for the performance of the process are developed. The process performance cycle is defined. Key milestones for the performance of the process are established. Estimates for process performance attributes are determined and maintained. Process activities and tasks are defined. Schedule is defined and aligned with the approach to performing the process. Process work product reviews are planned. **GP 2.1.3 Monitor the performance of the process against the plans.** [ACHIEVEMENT c] The process is performed according to the plan(s). Process performance is monitored to ensure planned results are achieved and to identify possible deviations **GP 2.1.4 Adjust the performance of the process.** [ACHIEVEMENT d] Process performance issues are identified. Appropriate actions are taken when planned results and objectives are not achieved. The plan(s) are adjusted, as necessary. Rescheduling is performed as necessary. **GP 2.1.5 Define responsibilities and authorities for performing the process.** [ACHIEVEMENT e] Responsibilities, commitments and authorities to perform the process are defined, assigned and communicated. Responsibilities and authorities to verify process work products are defined and assigned. The needs for process performance experience, knowledge and skills are defined. **GP 2.1.6 Identify, prepare, and make available resources to perform the process according to plan.** [ACHIEVEMENT f, g] The human and infrastructure resources, necessary for performing the process are identified made available, allocated and used. The individuals performing and managing the process are prepared by training, mentoring, or coaching to execute their responsibilities. The information necessary to perform the process is identified and made available. **GP 2.1.7 Manage the interfaces between involved parties.** [ACHIEVEMENT h] The individuals and groups involved in the process performance are

	determined. Responsibilities of the involved parties are assigned. Interfaces between the involved parties are managed. Communication is assured between the involved parties. Communication between the involved parties is effective.
Generic resources	**Human resources with identified objectives, responsibilities and authorities** [ACHIEVEMENT e, f] **Facilities and infrastructure resources** [ACHIEVEMENT g] **Project planning, management and control tools, including time and cost reporting** [ACHIEVEMENT a, b, c, d] **Workflow management system** [ACHIEVEMENT d, f, g]

Table 2.5: PA 2.1 generic practices and resources [5, Page 79-81]

Monitoring the performance of the process is very important in the context of this research work. The related generic practice GP 2.1.3 tells us to track and record the outcome of process and record any deviation. There should be some metrics defined before we start monitoring or measuring them. Key Performance Indicators (KPIs) are one way of defining metrics and then measuring them for the sake of monitoring. KPIs are discussed with more details in Section (2.3).

The second process attributes which needed to achieved largely for SPICE level 2 compliance is PA 2.2 work product management. This involves identification, documentation or definition of work products.

PA 2.2 is described in [5, Page 81] as follows:

PA 2.2 Work product management process attribute

The work product management process attribute is a measure of the extent to which the work products produced by the process are appropriately managed. As a result of full achievement of this process attribute:

a) Requirements for the work products of the process are defined

b) Requirements for documentation and control of the work products are defined

c) Work products are appropriately identified, documented, and controlled

d) Work products are reviewed in accordance with planned arrangements and adjusted as necessary to meet requirements.

Generic practices	**GP 2.2.1 Define the requirements for the work products.** [ACHIEVEMENT a]

	The requirements for the work products to be produced are defined. Requirements may include defining contents and structure. Quality criteria of the work products are identified. Appropriate review and approval criteria for the work products are defined. **GP 2.2.2 Define the requirements for documentation and control of the work products.** [ACHIEVEMENT b] Requirements for the documentation and control of the work products are defined. Such requirements may include requirements for (1) distribution (2) identification of work products and their components and (3) traceability Dependencies between work products are identified and understood. Requirements for the approval of work products to be controlled are defined. **GP 2.2.3 Identify, document and control the work products.** [ACHIEVEMENT c] The work products to be controlled are identified. Change control is established for work products. The work products are documented and controlled in accordance with requirements. Versions of work products are assigned to product configurations as applicable. The work products are made available through appropriate access mechanisms. The revision status of the work products may readily be ascertained. **GP 2.2.4 Review and adjust work products to meet the defined requirements.** [ACHIEVEMENT d] Work products are reviewed against the defined requirements in accordance with planned arrangements. Issues arising from work product reviews are resolved.
Generic resources	**Requirement management method/toolset** [ACHIEVEMENT a, b, c] **Configuration management system** [ACHIEVEMENT b, c] **Documentation elaboration and support tool** [ACHIEVEMENT b, c] **Document identification and control procedure** [ACHIEVEMENT b, c] **Work product review methods and experiences** [ACHIEVEMENT d] **Review management method/toolset** [ACHIEVEMENT d]

	Intranets, extranets and/or other communication mechanisms [ACHIEVEMENT b, c] **Problem and issue management mechanisms** [ACHIEVEMENT d]

Table 2.6: PA 2.2 generic practices and resources [5, Page 81-82]

Achieving level 2 means for each process performance management and work product management are applied in a fashion that risks and inconsistencies of bad planning are avoided. For example software integration testing (SWE.5 according to the Automotive SPICE identification scheme) the process performance management can involve monitoring number of passed and failed test cases thus notifying the management about quality and interoperability of units involved.

There are further levels of compliance which are discussed in details in official Automotive SPICE reference model. Here, the focus is only on level 2 and especially on process performance management.

2.3 Key Performance Indicators (KPIs)

Figure 2.6: KPIs collection cycle [8]

Key performance indicators (KPIs) are indicators which focus on those critical characteristics of organisational performance that are the most important for the current and future success of the organisation. KPIs are hardly new to the organisation. Either

they have not been known or they were sitting unused somewhere unidentified by the current management team. [9]

KPIs are adopted and used by almost all the industries. They might be different in different companies but serve the same purpose for example in a call centre customer satisfaction could be an important KPI where for an airline company on time departure and arrival are more important KPIs. In recent years, a big topic of discussion is that a high proportion in the optimization of production processes, the concrete organisational processes are the means of production.

Implementation of KPIs:

As part of the implementation of KPIs, it is first determined: Which business processes are investigated? The identified processes are then decomposed within the process chain in blocks. The individual modules can now be regarded as isolated steps within the process chain. Then measurements are carried out for each individual step.

The definition of KPIs is based on dimensions in which the measurements are defined during the identification of the operations. Here correlated variables are of great importance, as can be revealed by vital dependencies in the process chain.

Defining KPIs:

Another important part of KPIs implementation is KPIs identification. KPIs identification can be divided into three steps i.e. Get clarity and consensus, aligning goals and identifying what is important to measure.

Some other important points needed to be kept in mind while defining KPIs in an organisation are as follows:

- Should be measurable on defined frequency
- Oriented towards top level management
- Should not be related directly to currency e.g. euros, dollars, etc.
- The responsive action should be easily deducible from respective KPIs
- The performance of process should be related directly to its KPI (e.g., delay in last three milestones shows poor project management)
- KPIs should be related with responsible person, department or team. This is because in case of issues senior management can talk directly to responsible party

If appropriate benchmarks are defined for every KPI, in a final step, a monitoring tool is implemented. This secures an overall view of all processes. For monitoring Business Intelligence (BI) tools are being used widely these days.

BI is a vast topic, anyhow introduction to BI, examples of BI tools used in automotive industry and their importance is discussed in next section.

2.4 Business Intelligence

The term Business Intelligence became popular from the early to mid-nineties in 20th century. It refers to procedures and processes for systematic analysis (collection, evaluation and presentation) of data in electronic form. The solution used to implement these procedures and processes is referred as BI system. [10]

"This intelligence system will utilize data-processing machines for auto-abstracting and auto-encoding of documents and for creating interest profiles for each of the "action points" in an organisation. Both incoming and internally generated documents are automatically abstracted, characterized by a word pattern, and sent automatically to appropriate action points."[28, Page 315]

Modern BI systems are far more refined and advanced. Using BI systems is not processing documents anymore. But the basic concept is still gathering critical information about processes important for organisation's business and then presenting it to stake holders in meaning full way. In a narrower sense BI designates only the methodology of data collection.

In a broader sense Business Intelligence is the totality of management such as knowledge management or customer relationship management. It is also the permanent data maintenance and adaptation to a changing environment in a process oriented business model. The Institute for BI in Berlin defines "business intelligence" as the integration of strategies, processes and techniques to generate income from distributed and inhomogeneous business, market and competitor data knowledge about status and potentials and prospects.

2.4.1 Implementation of BI System

Exercising business intelligence also involves automation of controlling, reporting, planning, reviewing and analysis of market and customer. Figure 2.7 shows generic architecture of a BI system.

A BI system can only be fully effective if the data coming from the source systems are valid. An organisation dealing with many tools, having multiple dependencies of data or don't have up to date data, possesses constant threat of data chaos. The implementation of BI system can be divided into three phases:

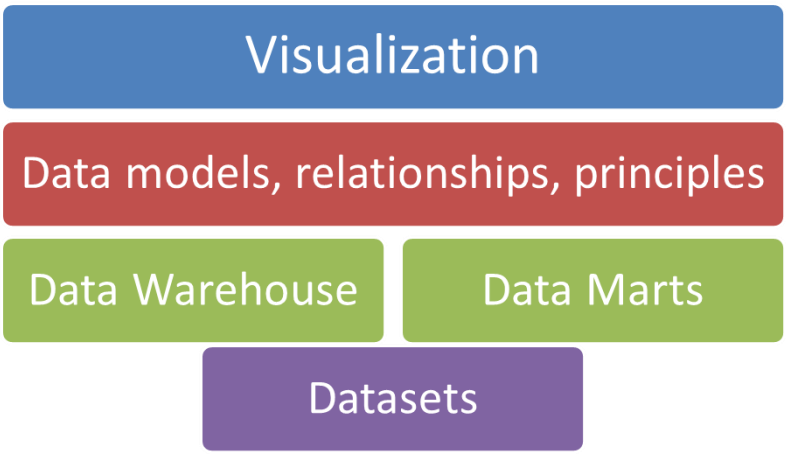


Figure 2.7: Generic architecture of BI systems

First Phase: Key data is collected (quantitative and qualitative, structured or unstructured). This data collection is either via operation system or from datasets. In most cases this data is stored in data warehouses or data marts. The process of extracting data from source systems and bringing it into the data warehouse is commonly called ETL (Extract Transform Load), which is extraction, transformation, loading of data. [11]

Second Phase: Relations, patterns, and principles are developed within collected data. This helps recognizing patterns and discontinuities.

Third Phase: Findings are presented to the company. The distribution of the findings serve basis of decisions, required measures and actions.

2.4.2 Business Intelligence Tools

Currently there are commercial and open source BI tools available in the market. More than 150 companies are providing BI solutions. Some famous providers of commercial BI tools are as Microsoft, IBM, Oracle, QlikTech, SAP and Tableau software.

Currently it is golden era for open source solution but yet open source BI tools need to get more mature as Information Technology (IT) giants are providing cutting edge customer service and updates. Some noticeable open source BI tool providers are BIRT, Pentaho and SpagoBI.

Modern BI tools are focusing more on data discovery which provides ease of use, flexibility and a lot of freedom for users. Data discovery tools also differentiate their objectives by analysing current problems and providing forecast of data to help analyse answers of questions about goals and targets.

Most BI tools don't target specific industry and can be adopted by different trades as required. There are some tools which provide different basic purpose but can also help in achieving certain standardizations by giving features like tracking and monitoring. Following are some examples explaining current trends:

- Audi AG is using Tableau in various departments for monitoring and tracking [12]
- Metro Bank is using Microsoft Power BI to track KPIs like call volume from call centres, number of transaction from mobile devices, customer dissatisfaction, reports, etc. [13]
- Küster Automotive GmbH produces apparatuses (breaks, exhausts, flaps, etc.) for the automotive industry. As most of their customers needed SPICE level 2 compliance, so they adopted application lifecycle solutions from Polarion to achieve it. Polarion doesn't explicitly produce BI solutions but their products are focused towards automotive industry. So they decided to include monitoring plugins which help in BI activities [14]
- Similar example is for Continental AG using Code Beamer from Intland Software to support their collaborative software development processes [15]

2.5 Datasets

A dataset refers to a larger, contiguous amount of data. Dataset despite verbal similarity is not to be confused with a record. A dataset can contain single to multiple types of data. Dataset not only refers to the database but it means the source of data, it can either be data mart, excel file or a software development tool.

Nowadays because of different skill level of staff, individual choices and time of adaptation, multiple tools are used within single organization. Another reason of adapting multiple tools and not going with single solution is sacrificing compatibility and easy interfacing for the sake of freedom of functionalities and latest solutions. To understand type of development environment in typical automotive supplier, tools used by Mentor Graphics Germany (where the research work has been carried out during this thesis) are discussed in following sub sections.

2.5.1 JIRA

JIRA is web based software for fault management, troubleshooting and operational project management. JIRA is used also in non-technical fields for the task management. It was developed by Atlassian. Key use of JIRA is in software development where it supports requirement management, status tracking and later

the troubleshooting process. JIRA's main role in an organisation is process management and process improvement. The name interesting is not an abbreviation and came from Japanese word "Gojira (Godzilla)" to make reference to its competitor Bugzilla. [16]

In Mentor Graphics JIRA is serving by sharing and visualizing tasks among teams with its basic concept of creating issues. An issue consists of a project assignment, summary, type, priority, component assignment, content, etc. An issue can contain much more information either through more fields, attachments or through added comments. The issues can be easily edited or changed to different state e.g. Open to Closed. Workflows are defined with the help of state transitions. Any change to an issue is also logged.

JIRA has a large number of configuration options: For any application, a separate issue type with own workflow, private status, one or more different status dependent views, own fields and manual or automated workflow transitions are developed. JIRA also has reporting system but for custom reports, free or commercial plugins are required. It is also not possible to import data from other sources for reporting across multiple data sets.

In Mentor Graphics issues are associated with different milestones. Completion of milestones can be determined by the status of associated issues. Still projects in JIRA don't have regular format and need to be more standardized. JIRA's flexible architecture enables the user's ability to develop extensions for JIRA and then publish them to Atlassian Marketplace. JIRA uses Representational State Transfer (REST) for interfacing with external tools and resources.

2.5.2 Excel Files

Microsoft Excel allows user to perform extensive calculations with formulas on commercial, statistical and date functions. Excel also has many math functions, so that many problems of business mathematics can be calculated. It can concatenate text and also logical calculations can be carried out.

Even in modern software development processes Excel files are extensively used. The functionality and application of Excel can be extended by programming in Visual Basic. The test plans for projects and their results in Mentor Graphics are stored in Excel file. Using Visual Basic scripting this test plan is also connected to a My Structured Query Language (MySQL) database in the back end. Moreover the Technical Requirement Specification (TRS) has also Excel template.

Modern BI software providers are familiar with the fact that Excel like software are widely used among multiple processes. Almost all the modern BI solutions

provide out of the box connectivity between excel files and visualizations. The problems with using excel files are they are not always up to date and manual data entry is required in most cases.

2.5.3 OpenAir

NetSuite's OpenAir is professional services automation (PSA) solution. The key intended purposes are resource management, project management, time and expense tracking, project accounting and advanced billing and Invoicing. [17]

Mentor graphics uses OpenAir in many aspects, but interesting features for Automotive SPICE compliance are timesheet management, expense management and resource management. Management and stakeholders of project are really interested about on time delivery and progress. By gathering valuable data from OpenAir management can realize problems in resource allocation or work distributions.

OpenAir has good integration with current office and Enterprise Resource Planning (ERP) systems. OpenAir uses XML API for interfacing with other systems or tools.

2.5.4 Jenkins

Jenkins is an extensible, web based system for continuous integration of components into an application program. It is free software released under Massachusetts Institute of Technology (MIT) license. In Mentor Graphics it is used for continuously building and testing projects. Every project is allocated to a separate server and a single server can handle multiple jobs.

The output of these builds help gathering valuable Static Code Analysis (SCA) data using FlexeLint. Separate jobs can also be defined for the purpose of testing other than basic testing which is part of builds. Dependencies between jobs can also be defined. Jenkins too uses REST based API.

There are many more data sets available in environment, which are not directly related to this research work. One of the major goals of designing BI system for certain standard achievement is to make it flexible so that it stays scalable and can help attaining higher standardizations in future.

Summary

Standardization in Automotive Industry is becoming more and more important. In past generic software standards were enough for software development in automotive industry. With increasing number of EE and software systems, specific standards needed to be developed out of these generic standards. For example OSEK and AUTOSAR are standards for architecture of operating systems used in ECUs. Automotive SPICE is decedent of ISO 15504 which addresses best practices, tracking and monitoring of processes during software development.

Automotive manufacturers are facing challenges like increased reliability requirements with increase in complexity, short time periods for product roll out, using legal components and assuring the interoperability of components from different suppliers. Many automotive manufacturers are demanding Automotive SPICE level 2 compliance from their suppliers. Process performance management and work product management are two main attributes of level 2. For the purpose of tracking key processes and monitoring their performance KPIs can be defined across different software development processes. These KPIs not only shows how well process is being carried out but also help the management to take relevant actions and discuss issues with respective department or team. For effective monitoring and tracking of process performance, these KPIs needed to be collected automatically. Business intelligence systems are great way to collect and visualize KPIs required for level 2 compliance. There are open source, free and commercial BI systems available.

Before starting the implementation phase it is important to understand software development norms, practices and tools used by automotive suppliers. Understanding of datasets and their interfacing capabilities is also vital. With different tools used among different suppliers presenting a standard solution for tracking and observing progress of process performance is almost implausible.

Chapter 3

3. State of the Art

Attaining fast reaction times to moving market conditions and regulation, and simultaneously keeping up high quality standards for products, companies must have reliable and repeatable methods and processes. For the development of software and electronic systems Automotive SPICE is the state of the industry model, which nowadays is required by almost all major European car manufacturers. [18] This chapter explains the state of the art. Basic concept of SPICE and Automotive SPICE are already presented in Section (2.1). Here importance and trends of Automotive SPICE are discussed followed by possible current situation of the development process in Mentor Graphics. Later this chapter discusses some proposed approaches for KPIs collection system in past, famous tools available for automotive industry which can be used for monitoring and issues in implementation of these approaches.

3.1 Importance of Automotive SPICE

Automotive SPICE has already established itself in all areas of ECU development as a process reference model. It takes long time for automotive supplier to achieve stability at certain level of SPICE compliance and then years more to make transition towards next level of compliance.

In Automotive SPICE process assessments, OEMs evaluate process maturity level of their suppliers. Maturity levels in combination with information about technology, quality, cost and infrastructure are the foundation of the manufacturer's internal ratings for suppliers. Depending on these assessment results, automotive suppliers are classified internally. Bad assessment results effect straight away in the drafting of suppliers and possibly may even lead to loss of supplier status.

Currently OEMS usually expect Automotive SPICE level 2 from their suppliers. For supplier Automotive SPICE assessments are almost compulsory before moving forward with OEMs. It is therefore in the favour of suppliers to seek and give extra time for assessments of SPICE to save the time during assessment window. Only with successful assessments there is a change for suppliers to enter in competition presented by the innovation age in coming future.

The important question here is that, Is Automotive SPICE suitable as an assessment model? The answer is not quite simple. By definition Automotive SPICE is a process model, which is a composition of important practices and guidelines for software development in the automotive industry. It is a method for determining the maturity of

the defined processes. It is not an average assessment score of the maturity of an organisation or project; rather it is related to individual processes, such as engineering processes or risk management by rating them on scale from zero to five.

For projects under development Automotive SPICE is more about meaningful evaluation of the individual disciplines, with the associated strengths and weaknesses. This detailed statement of the actual status of a specific software project makes Automotive SPICE the ideal starting point for process improvements.

Automotive SPICE is more than just a model for the assessment of processes. The way how each process is described by the best practices and usage of resources, it is expected that Automotive SPICE can also be considered as a guide for high quality software development. But the progress can only be noticed if procedures and practices are implemented as defined in reference model.

During the implementation phase of Automotive SPICE users can be classified into two groups:

The first group covers assessors who utilize the inventory developed during the implementation phase and use Automotive SPICE as a tool to determine maturity level.

The second group improves quality and carry out practices and guidelines defined by Automotive SPICE.

The purpose of Automotive SPICE depends heavily on the timely progress in the project under development and on the particular user group. Most of time results of assessments often trigger the input parameters for internal improvement measures but as a premium supplier prerequisite for a successful review and qualification should be ready to be the part of consistent internal communication with customer.

When interpreting assessment results, it is often the problem of which type of assessment to be use. The customer who wants to carry supplier’s project evaluation is faced with the challenge of whether to use the results of already carried out assessment from supplier or use its own assessors. On the other hand, a supplier organisation that has appointed an internal quality assurance team is confused to either use an external service provider for assessment or not, to make sure whether its results will be reliable with the possible review by a customer. The assessments performed by customer tend to have lower ratings than those which have been carried out as part of internal improvement measures. Inexperienced assessors tend to measure better, since they see less evidence of the processes. Thus going with an external certified evaluator is better for automotive suppliers.

Automotive SPICE is not only beneficial for certification or to show customer certain maturity level. Implementation of Automotive SPICE can also lead to some benefits discussed in following sub section.

3.1.1 Benefits of Automotive SPICE

Other than benefits defined above, Automotive SPICE has some really good impacts on software process development. According to [18] an organisation can take benefits from Automotive SPICE in following ways:

Efficiency improvements can be achieved while moving from one level to another.

Defect reduction in software development depends upon approach used by organisation. At level 2 and level 3 defect reductions can easily be observed which shows a positive effect on the organisation.

Productivity gain can be realized as per capability level is achieved. This is because of low effort spent on bug and defect fixing.

Increased predictability of projects can also be achieved because of availability of past data from previous projects the organisation has completed. Predictability is mostly associated with higher levels of Automotive SPICE.

3.1.2 Current Trends in Automotive Industry

In practice Automotive SPICE is part of the quality management system for OEM and ECU suppliers. For OEMs it is concrete purchasing strategy for supplier evaluation and also a part of the internal software development strategy for the monitoring of different safety critical projects. Embedding of supplier assessments according to Automotive Spice under a highest maturity level is useful in many cases. The aim is to determine the entire chain of the development cycle for each component, from the first impression to series production process maturity of the relevant supplier.

On the supplier side a concerted strategy for software based system development is usually already established. Other than Automotive SPICE, results from other assessment models are also included in report. For example CMMI reports, for the functional safety of safety critical systems ISO/IEC 61508 and in particular its automotive-specific implementation in the upcoming ISO/IEC 26262. [19]

In all these accomplishments it must not be forgotten that a long term success of Automotive SPICE will ultimately occur only through the industry wide implementation of a defined and promising approach. Goal of a long term strategy

must be therefore, to precisely enable users through training process for the improved implementation of Automotive SPICE.

3.2 Requirements of a KPI Monitoring Tool

This section describes the key requirements for KPI Monitoring Tool for an Automotive Supplier in functional and non-functional groups. These requirements are gathered during research work in Mentor Graphics and may vary for different suppliers accordingly.

3.2.1 Functional Requirements

Refresh Frequency: System should be able to update data on request. Also there should be functionality for refreshing data at pre-defined intervals.

Graphical Representation: It is key part of KPIs collection systems to show data graphically. Other than achieving certain goals for monitoring and tracking in compliance process, this tool is intended for higher level management, so it is necessary to get meaning full and user friendly graphical representations of data.

Flexibility: Flexibility is always help full in ever changing era of technology. KPIs shouldn't be hard coded and user should be able to add new KPS with little to no effort. Users should also be able to customize layouts, edit current metrics and import/export data in different forms. It is also expected that the system should not only grab data automatically from various sources but also provide user with ability to insert data manually.

Extensible: Extensibility can be related with future expansion of system. It is always considered good when system uses latest APIs and standards to integrate and communicate with current and upcoming tools used in software project development and management.

3.2.2 Non-Functional Requirements

Security: The system should be able to use organisation authentication domain or closed domain which can be administered by organisation's IT team.

Privacy: Data related to projects is really sensitive. It is intended that this data should not be published or leaked to public in any case. Best case scenario is to not publish data externally but if data is used outside the organisation premises privacy and security of data should be taken care of.

Robustness: The system suggested in this thesis must be able to communicate with almost all the tools used in software development tools used in automotive

industry. That is why evaluation of available approaches is big part of this research work.

3.3 Current Situation

In this section, first current situation is discussed from perspective of project monitors i.e. process and quality, product delivery, test and project managers and the higher level management. Then current development process is presented towards the end of this section.

Process and Quality Manager:

The responsibilities of process and quality manager are to analyse current process models, present new ideas for process improvements and evaluate implemented process model for certain level of acceptance and compliance. The research work is carried out in an organisation which has successfully introduced Automotive SPICE at level 1 and is working on level 2. From a standard and business perspective, it is necessary to collect main process and project. Currently, this is done manually and not in every project. Referring to continuous process and acceptance improvement, it has become mandatory to automate the collection of most of the KPIs. The KPIs are already defined but more KPIs could be defined or changes can be made to current KPIs. Towards the end of implementation of KPIs collection system process and quality manager will once again evaluate the approach and suggest further steps needed for the improvement of system.

Product Delivery Manager:

Product delivery manager deals with delivery of projects. Billing milestones, version releases, managing planned working hours on a project and coordinating with project manager for onetime delivery of project are some of major duties of a product manager.

Billing milestones are time stamps where customers are paying for the work done. Billing milestones are really important and a project manager can tell about delay or potential issues by looking at past billing milestones. In current environment OpenAir is used for this purpose. The problem with OpenAir is that it only shows the latest billing mile stone date. To check future or past dates, manager has to go through a lot of hassle. So it is required to draw data from all billing milestones to present it as a KPI in implemented system.

For developers to work on a project, they need to book working hours by sending request to project manager. When a developer books hours they are assigned to respective project after all his/hers requests are accepted by project managers from different projects. So if not all managers approve bookings, the managers who partially

have approved hours will see more approved hours than planned. By drawing a KPI which differentiates between initially planned hours and current hours can solve this problem.

Test Manager:

Test engineers perform all the test cases which are defined in a test template and results are updated automatically in main page of test template. Initially there is only test template available but a lot changes are going in testing department. Automated testing is also not completely implemented yet.

Initial requirement of testing KPIs were to grab data from excel template. Afterwards a MySQL database was implemented. This database is more up to date and overcomes the problem of locating test template of each project separately. KPIs about Static Code Analysis (SCA) are mostly available using Jenkins.

Project Manager:

Project manager usually need more deeper and technical knowledge about projects. KPIs with more quantitative approach are interesting for a project manager. Still project manager is interested in overview of his/hers projects to quickly realize which department is facing more problems.

Project manager is more interested in flexibility of system. It is required for system to go for a wide reporting system where project managers can create their own dynamic reports thus without spamming the reports intended for stakeholders with less technical knowledge about project.

Higher Level Management:

Higher level management is responsible for making decisions about taking in more work, assigning budget to particular area or reducing current work load. The higher level management is not using any tool for KPIs collection. Currently no support system is available for decision making. To get overview of project higher level management need to communicate directly with project managers or rely on bi-weekly meetings. Higher level management is more interested in a tool which can help them in effective and quick decision making and not interested in micro details.

The goal is to present data in a meaningful way. There are not so many users involved at higher level management, so the visualization of reports and KPIs can be pre-assembled.

Current Development Approach

Currently office tools from Microsoft are used excessively in development life cycle. The organisation is following modified version of V-Model for process. The possible current approach can be seen in details in Figure 3.1. Following tools were used before implementation of level 1 SPICE compliance:

Acquisition, Project Management:

Microsoft (MS) Word is used for quotation in in acquisition where Microsoft project is used for project planning and Project Management Professional (PMP). MS Project is also used for macros used by excel Technical Requirement Specification (TRS) template.

Requirement Engineering:

There are currently no guidelines available for customers to use specific tool. It is better to give customers freedom to use tool of their choice. Locally MS Excel TRS template is used for requirement gathering and approval.

Software, Hardware and FPGA Design

Unified Modelling Language (UML) is used for designing software specification inside Enterprise Architect. For hardware design Altium Designer is used. HW and field-programmable gate array (FPGA) designs are reviewed and documented in MS Word and MS Excel.

Software, Hardware and FPGA Development:

There are several software and hardware development tools available. Using development tools depend on the projects. Usually suppliers are working on the part of the project so it depends upon the customers, that what kind of programming language they are using in which environment. For pseudo coding or comment-driven development and reviews again MS Word and MS Excel are used.

Integration, Verification and Testing

Jenkins is used for continuous integration on dedicated servers. Parasoft is used for SCA. MS Excel template is used for test plan and documentation in combination with MS Word.

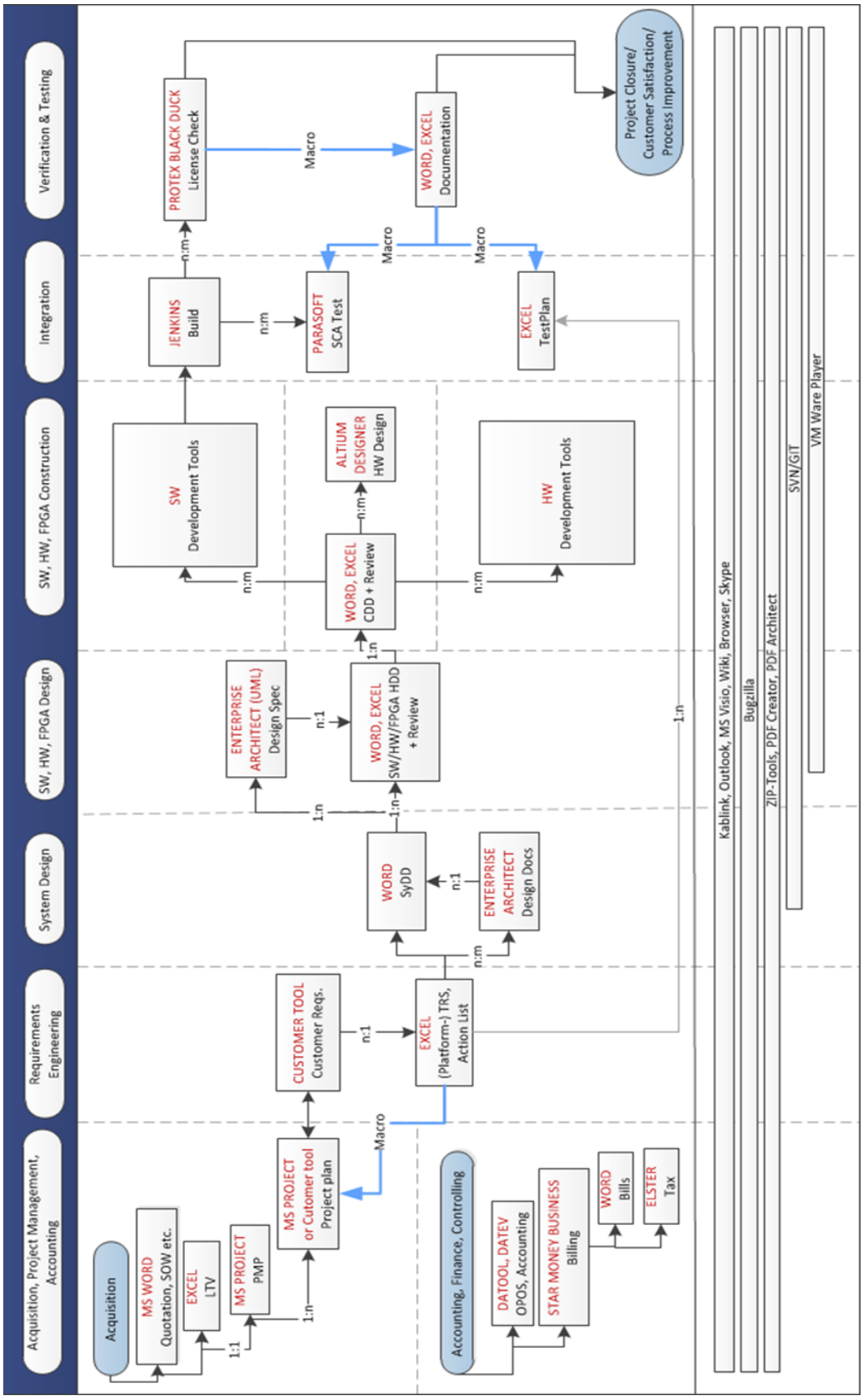


Figure 3.1: Possible development approach for an automotive supplier [20]

Supporting Tool

Some supporting tools worth mentioning are Kablink, Outlook, MS Visio, Wiki, Skype, Bugzilla, Zip Tools, PDF Creators, Virtual Machine Ware Player, etc.

For level one SPICE compliance some new tools were introduced. JIRA played major role in combination with SharePoint, confluence and stash.

3.4 Previous Proposals

As easy as it may seem achieving level 2 SPICE compliance but it is quite challenging. Mentor Graphics automotive business unit has previously tried to implement KPIs collection system. This section discusses briefly about approaches proposed by previous proposals and their strengths and weaknesses.

3.4.1 First Proposal Microsoft Office with Scripting

This approach has low initial effort with the need of a web developer.

- Start with MS Office

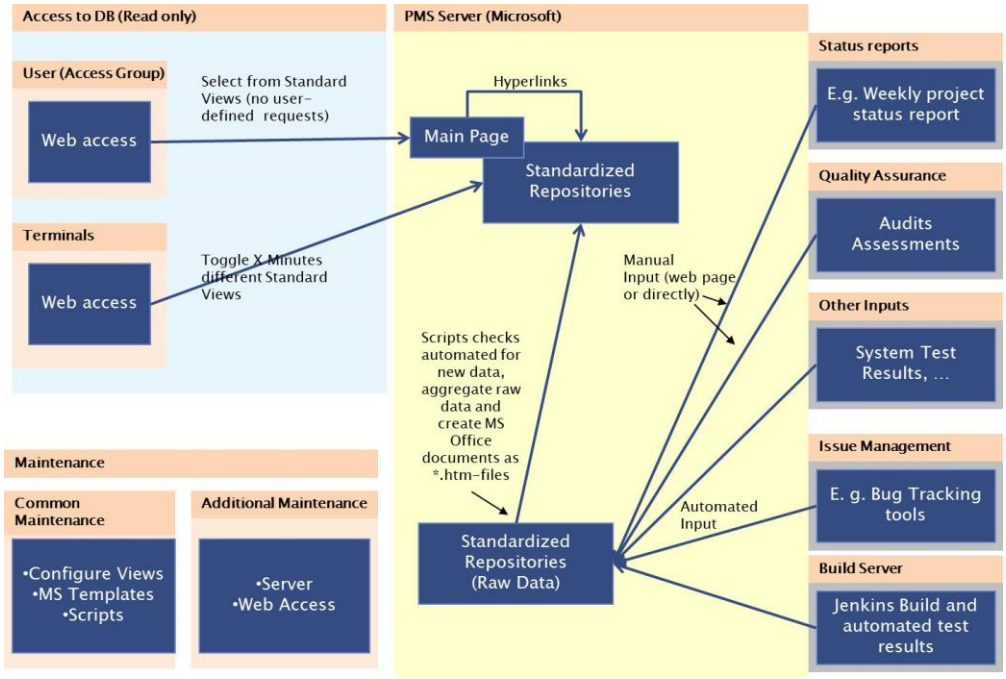


Figure 3.2: MS Office base KPIs collection System [21]

- Get experience about what kind of data is needed and how it should be aggregated
- Switch to database approach once Performance Measurement System (PMS) has reached on a stable condition

Pros:

Few assistance of engineering at beginning needed, Goal to visualize Indicators fast to achieve and internal and external communication may benefit directly from this solution.

Cons:

More effort, Microsoft (MS) Office approach will become out of fashion quickly with no reuse.

3.4.2 Second Proposal Database with Scripting

This approach requires high initial effort with the need of engineers to develop relational database model, database and web portal.

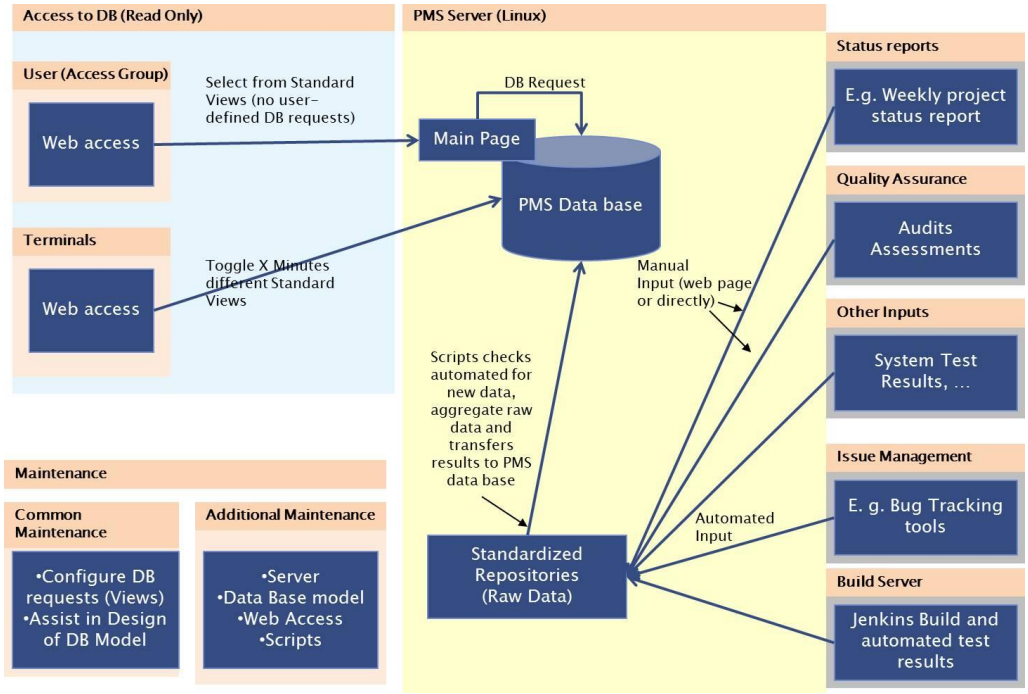


Figure 3.3: KPIs collection system based on centralized database [21]

- Start with database
- Start with almost no relational database model
- Get experience about what kind of data is needed and how it should be aggregated
- Reconfigure database model once performance measurement system has reached some kind of stable status

Pros:

Concentrate efforts on final solution, easier maintenance in mid to long term usage and less effort in mid to long term usage.

Cons:

To build and implement solution even on example project, effort and resources needed are too much.

The reasons for these systems never got final approval because both proposals needed to be build-up from scratch. Even first live demonstration needed a lot of resources and visualization capabilities of both proposals were quite limited and out of fashion. Scalability and flexibility were big concern for higher management.

Summary

Automotive SPICE can be used both by OEMs and automotive suppliers. OEMs perspective of using Automotive SPICE is to improve internal software development process and to use it as an assessment tool for evaluation and categorization of suppliers.

Automotive SPICE is not just a process reference model used for ratings of process in form of capability levels but it can also be used as a guide for high quality software development in safety critical systems alongside with ISO/IEC 61508 and ISO/IEC 26262. Currently OEMs are relying on Automotive SPICE ratings while sourcing out of their projects partially. On the other hand suppliers are focusing more and more to attain higher levels on Automotive SPICE capability scale. Automotive suppliers are not only auditing internally but also inviting experts to evaluate their current practices in software development processes. This is due to persuasion of higher score when they are assessed by OEMs.

OEMs usually expect minimum of level 2 Automotive SPICE compliance from their suppliers. As tracking and monitoring is big part of level 2 compliance, it is recommended to capture KPIs for projects. A KPIs collection system should have good graphical representation capabilities with scalability and flexibility. System should also be able to work within secure environment while still be able to communicate with external resources. Different managers have different requirements for KPIs. System is intended mainly to help achieving level 2 Automotive SPICE compliance and to give critical information to higher level management.

Achieving level 2 Automotive SPICE compliance is quite challenging and implementing a monitoring system is quite new idea for automotive industry. Failure is usually caused in implementation schemes due to absence of knowledge about possible approaches, lacking technical expertise and fear of big investment and little gain from higher management.

Chapter 4

4. Evaluation of Approaches

This chapter discusses about available approaches to gather and present KPIs for available projects. In start of this chapter, discussion about adoption of Application Lifecycle Management (ALM) solution from Polarion is presented. It provides some famous development tools which are used by automotive suppliers. Then brief discussion about building in-house solution from scratch. Towards the end of this chapter free and open source BI platforms are discussed in comparison with commercial solutions. Architecture, functionalities and evaluation of commercial BI tools are presented at the end.

4.1 Polarion ALM

The aim of Polarion ALM is to help OEMs and automotive suppliers with one integrated solution that brings project transparency through aggregated management information while moving towards compliance in famous automotive standards. Like any other ALM solution it helps different departments to react quicker to needs of potential and already acquired customers. [22]

Tracking with Polarion:

As shown in Figure 4.1 tracking can be carried out throughout the development life cycle with Polarion reporting. Polarion reporting allows users to use Velocity scripts in Wiki pages, which helps accessing any Polarion data through open API and extracts the desired data.

According to [22] with Polarion, users can easily customize and create different kind of reports, create cross domain and cross project reports and plug in additional business logics to measure KPIs which are explicit to the environment.

Advantages:

Following advantages of Polarion ALM are stated by [22]

- Project templates specially designed for automotive OEMs and suppliers can make process assessment and functional safety compliance easier
- Model Driven development

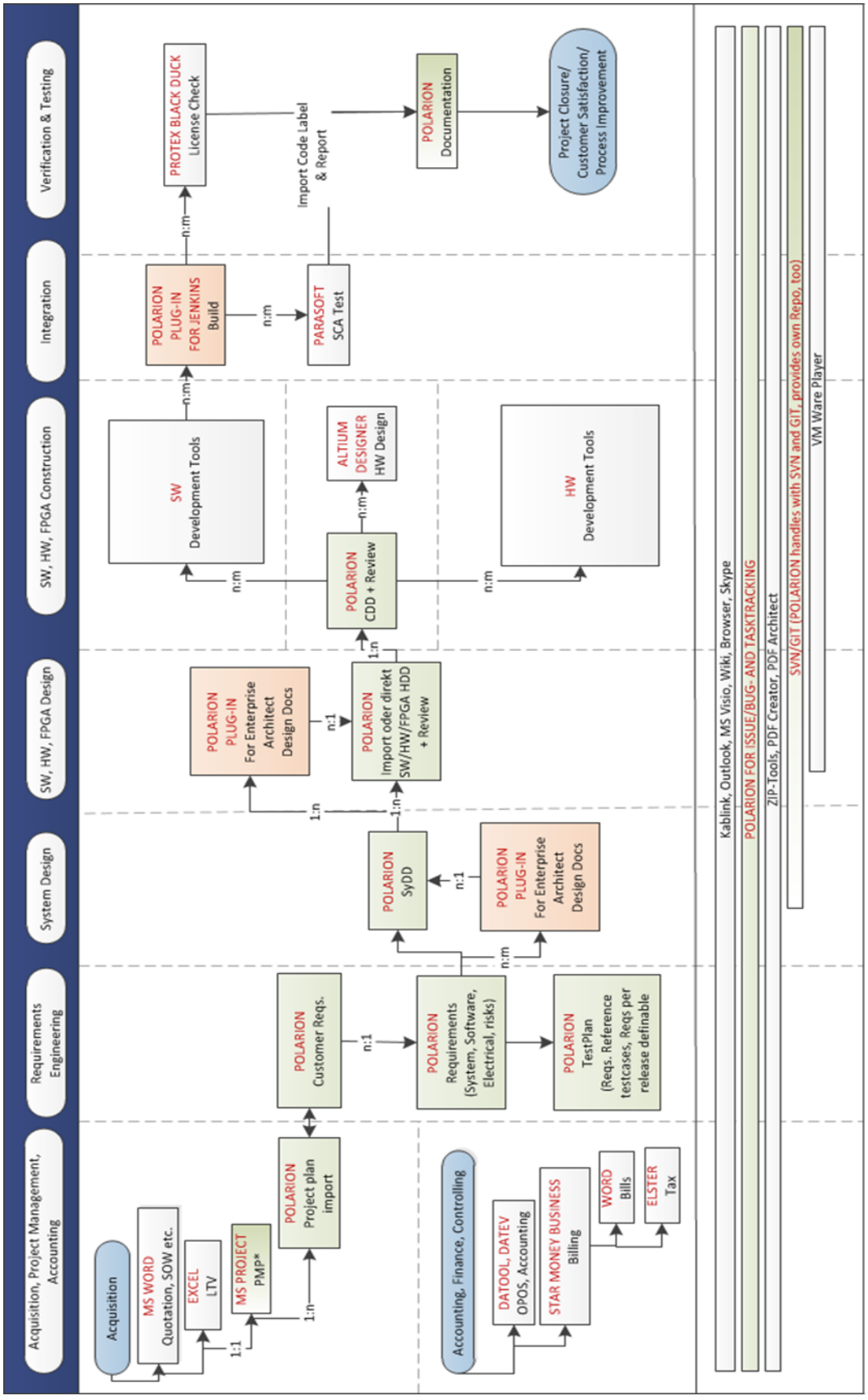


Figure 4.1: Implementation of Polarion in possible current situation [20]

- Traceability and tracking help in functional safety and compliances.
- Continuous real time collaboration
- Data exchange capability with common document formats such as Word/Excel or PDF

Disadvantages:

It is neither intended nor purpose of this thesis to find bad practices or functionalities in available tools. Following are rather disadvantages of using Polarion ALM in current development environment.

- Adaption of Polarion involves a remarkable amount of work. Other than analysing current environment, configuration, testing, the tailoring of resources is required
- High price is involved in adoption process. The price includes buying Polarion and price of migration process from legacy tools, which is quite high
- Flexibility and scalability are somehow compromised as compared to using best tools for particular processes

A similar solution used by several automotive suppliers is available from methodpark.

4.2 In-House development

In house development means building software from scratch using internal resources. As discussed in Section (3.4) building a solution from scratch is more suitable for organisation's customized needs but usually the efforts and amount of resources needed, can easily overwhelm the benefits.

Advantages:

- In-House development approach can help to map individual processes accurately
- When requirements are not well understood, adoption of a complex commercial solution is more expensive and more risky than comparable in-house developed solution
- Adapting a commercial solution comes with extra price of training and with the need of external support. This cost can be minimized as technical staff for in-house developed solution is more familiar with it.

- For in-house developed solution organisations can define their own data privacy and security rules, so they don't have to worry about safety, legalizations and security measures offered by commercial solutions

Disadvantages:

- Longer development times in fast growing innovation era can lead to out dated final solution
- If organisation's main focus is not the solution under development, the in-house developed solution can lag severely in quality and functionalities in comparison to its commercial counterpart e.g. an automotive Electronic Control Unit (ECU) supplier working on an Enterprise Resource Planning (ERP) system
- Support and training costs may seem low but change in key development staff can lead to bigger problems
- Hidden costs in terms of development resources and working hours can easily exceed the expected cost
- Commercial solutions implement state of the art encryption and authentication methods which are difficult to implement for in-house developed solution

The question of whether an organisation should develop in-house solution or adopt commercial solution is still one of the topical questions. Both approaches should be considered in conjunction with each other, with focusing on the applications of intended solution.

The main reasons KPIs collection system is not been developed from scratch during the course of this thesis, are time constraint and limited resources. It is not possible to gather all the resources with development, implementation and then test in-house solution during the course of six months.

4.3 Business Intelligence System Based Approach

Introduction to BI is already presented in Section (2.4) along with names of some famous free and commercial BI solution providers. Automotive SPICE does not recommend use of any particular tool or method for tracking the performance of processes. Different OEMs and suppliers use different techniques and tools at certain levels or processes. Moreover Automotive SPICE assessment reports are highly confidential, so there are few case studies available for research purposes.

After evaluation of ALM and in-house development approach it is been concluded that both approaches are not suitable in current development environment. Adapting a BI

system is a perfect mix of using off the shelf tools with flexibility of developing add-ons or extensions according to requirement specifications. The BI solutions proposed here can be categorized into open source or free and commercial solutions.

4.3.1 Open Source Widget Based Web BI Solution

BI solutions are somehow related to web technologies. This is because the visualization should be available to multiple platforms. Web topologies are highly cross platform compatible, which allows reports to be deployed to different devices and operating systems.

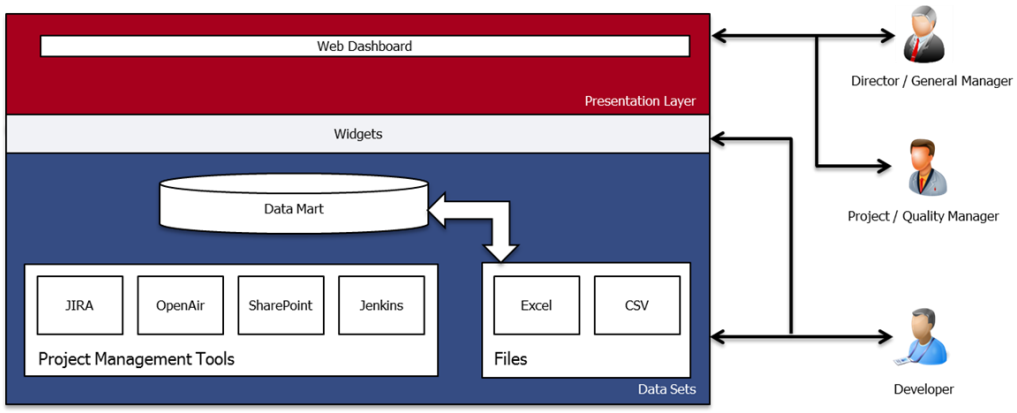


Figure 4.2: Architecture of open source BI solution [23]

Most web topologies like HTML, CSS, JavaScript, etc. can be used easily by different development tools. That is why most famous open source BI projects use these web technologies e.g. BIRT, JasperReport and Penthao are Java based open source BI solutions.

Open source BI solutions can be implemented using different approaches and architecture. Figure 4.2 shows generic architecture for implementation of open source web based BI solutions in current development environment. Most open source BI tools lack in providing comprehensive visualizations. Managing dashboards and creating reports with exclusive access are also challenging tasks. The widgets layer depends upon kind of BI tool or technology used. The data set layer does not depend upon BI tool used. Data set layer is associated with software solutions used by the organisation to gather data for projects.

4.3.2 Commercial BI Solutions

First market overview run on BI tools reveals few famous BI solution providers like IBM, HP, Microsoft, Qlik and Tableau. BI tool development is new trend for industrial giants and currently Amazon has also entered in BI tool development

field. Other than famous providers there are hundreds of medium to small range companies providing commercial BI tools.

It is impossible to carry out evaluation on every commercially available BI solution in a time frame given for this thesis. After several meetings with higher management and using confidential data provided by industrial partners, MS Power BI, QlikView and Tableau were short listed. Tableau was later dropped because of overlap of functionalises with QlikView.

QlikView:

QlikView is the main product of the company Qlik. QlikView is an analysis and reporting system. The associative in-memory search technology evaluates data while providing the user insights and understandings of business data.

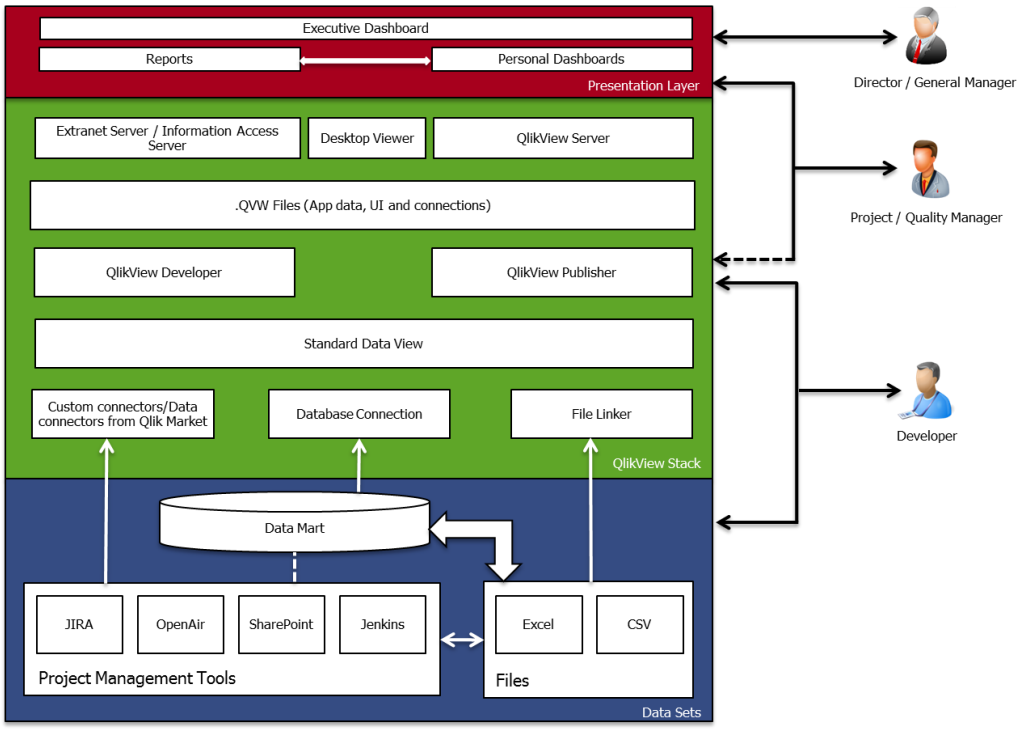


Figure 4.3: QlikView deployment architecture

QlikView provides analysis of information that is held in memory (memory-based analysis). Data from different sources can be linked together and are visualized in graphs and tables. Data storage in QlikView is based on a multidimensional database that is stored in main memory and provides OnLine Analytic Processing (OLAP) functionality. [24] As shown in Figure 4.3 the implementation of QlikView consists of three major layers. Each layer plays an important part in deployment of QlikView.

The presentation layer consists of web based dashboards and reports, developed using QlikView Developer (Windows based desktop tool used by designers and developer to create visualizations and logics behind reports). Administrator of QlikView server can manage the access of these dashboards and reports.

The middle QlikView stack layer consists of file linker, API and database connectors for interfacing. All the data collected is then converted into standardized view which in turn is usable by QlikView developer. Using QlikView publisher reports and dashboards are distributed on to internal protected server as well as to an extranet server for customer oriented reports. The standard file format used by QlikView is QVW which contains all the business logic, visualization templates and data needed for visualizations.

Data set layer is independent to QlikView stack. Qlik market provides commercial plugins as well as Software Development Kit (SDK) to develop plugins for data interfacing and several other functionalities.

Microsoft Power BI

Power BI is cloud based BI solution from Microsoft. It integrates really well into Microsoft ALM. Because of recently entering in BI market, it only provides cloud based solution at the moment but the future road map is to integrate Power BI with Structured Query Language (SQL) Server 2016 and SQL Server Reporting Services (SSRS).

Power BI developer and users can make use of two applications. The first is Power BI Desktop, a graphical data assessment and reporting tool. The second is a set of intuitive, collaborating mobile applications for Windows, iOS, and Android devices, providing secure access to live Power BI dashboards and reports from any device. Functionalities of Power BI can be extended with a set of REST APIs, a JavaScript framework which enable developers to customize Power BI to address distinctive needs of organisation. [25]

As shown in Figure 4.4, the presentation layer is quite similar to QlikView. The major difference is quality of graphics and interaction between different elements. Power BI uses state of the art latest graphical representation to draw charts and graphs compared to old and legacy html based visualizations used by QlikView. A sense of modern simplicity and tidiness can clearly be observed in Power BI visualizations, which makes reports and dashboards very easy to understand.

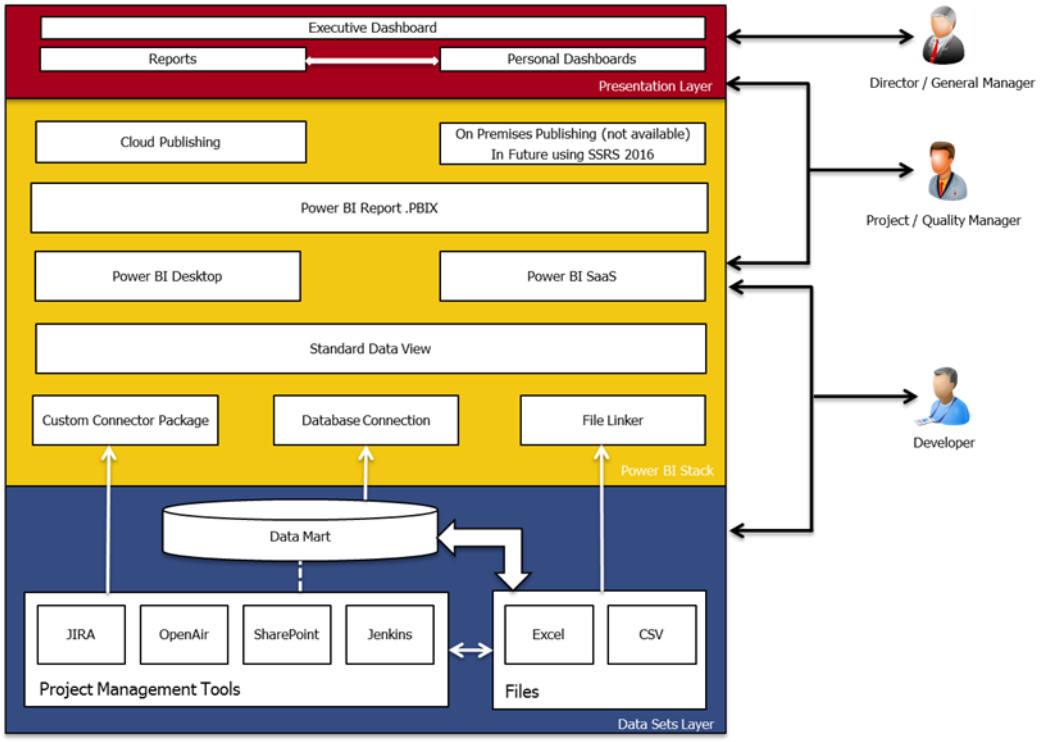


Figure 4.4: Microsoft Power BI deployment architecture

In Power BI, not only dependencies can be created within KPIs but also natural query language can be used for ad-hoc reporting. In Power BI stack layer, out of the box data connectors are available for all MS office products. Database connections are also available without installation of specific plugins like in QlikView. Custom connectors can either be built using .Net platform or REST API can be used if possible. Another advantage of Power BI is usage of Power Query which is widely known among MS office users. The view of collected data can be transformed using Power Query as needed. These transformation steps are not dependent on resources underneath and these steps stay intact on data refreshes. Further modifications and addition or deletion of unwanted data is possible using Data Analysis Expressions (DAX). A desktop editor or online editor can be used to create reports from transformed data and then publish these reports on online Power BI portal. Published data can be viewed only by people inside organisation domain. In future these reports and dashboards will be available on premises using SQL server 2016.

Data layer is similar to QlikView and does not need any special transformation for the deployment of Power BI. The interfaces in data layer are also independent of Power BI. Anyhow the API interfaces should be both supported by BI system and dataset to establish easy and reliable connection.

4.4 Comparison of Approaches

There are variety of BI system approaches and tools available. It is difficult to say which one is better overall. Adaptation of open source solutions will increase with time, but is it really for everyone and suitable for all type of software? On the other hand commercial software may seem easy to adopt but understanding the offered functionalities in commercial solutions is a challenging task.

Key Points	Power BI	QlikView	Open Source
Licensing Model	Storage Based	Token System/ Session Based	Free
Pricing	Free up to 1GB per user or 10 GB for 9.99$ per user monthly	1000$ per token	Free
Available in inventory	Not available	Not available	Not available
Learning curve for Data Analysts	Easy to Medium / Straightforward for office Power Users	Hard / Knowledge of Scripting languages is required to create good reports	Web Developer required
Suitable for types of Businesses	Mid-Market	Enterprise	Small Business
Support and Learning	Dedicated online support + Community	Dedicated online support + Community	Community

Table 4.1 Important features for implementation of BI solution

Table 4.1 shows important points for mid-size automotive supplier while implementation of mentioned approaches. The advantage for Power BI is, showing the

final solution before even paying a dime. Where for QlikView, server license is required before reports can be published. Open Source solutions are usually not suitable for agile development methods.

Requirements	Power BI	QlikView	Open Source
Dashboards for Mobile Devices	✓	✗	✗
Excel Connector	✓	✓	✗
Ad hoc Reporting	✓	✓	✗
Scalability	✓	✓	✗
Cloud / Extranet Publishing	✓	✓	✗
Natural Query Language	✓	✗	✗
API Connectors for present Tools	✓	✗	✗
Active Directory Authentication	✗	✓	✗
Partially without cost	✓	✓	✓
Available in MG Inventory	✗	✗	✗
On premises deployment	✗	✓	✓

Table 4.2: High level requirements checklist for proposed approaches

Table 4.2 shows high level requirements and either they are met in proposed approaches or not. The comparison clearly goes in favour of commercial solutions within current environment. The amount of requirements satisfied are also high by using commercial solution, so it is highly recommended adapting a commercial solution within current environment. This comparison is for the development environment of Mentor Graphics automotive business unit and should not be considered as an absolute comparison of BI solutions.

Summary

Developing BI tool from scratch is out of the scope for this thesis because of time constraints and limited resources. Open source widget based web solution is cost effective but less interactive and only provides short time solution. Selecting API calling, establishing data mart or both for interfacing depends upon selected BI tool.

QlikView is recommended for immediate on premises implementation. It has SQL like scripting but still extra training is needed for developers. Power BI is more user friendly, cost effective, suites skill set of most of people and has very good roadmap for integration with other Microsoft reporting solutions. Future roadmap for Power BI contains all the missing functionalities which are currently only available in QlikView.

Implementation will be carried out on Microsoft Power BI.

Chapter 5

5. Design and Methodology

This chapter proposes architecture for automated KPIs collection system based on Microsoft Power BI. It also builds up a methodology by considering proposed approach in Section (1.1) and mentioned requirements in Section (3.2). First half of the chapter discusses the architecture of the Microsoft Power BI based KPIs collection system. Second half explains the methodology by using it on an example project from major tier 1 supplier. This architecture and methodology is used for the implementation in following chapter.

5.1 Architecture of KPIs Collection System

The KPIs collection system is developed to automatically collect KPIs according to predefined logic. In proposed architecture this logic is implemented using development tool provided by BI system. The automation part is covered with schedule data refreshes. Some BI systems provide real time data update, but this can cause data security issues with extra costs e.g. storing data on cloud.

Figure 5.1 shows generic architecture of KPIs collection system. It is based on BI system and consists of three layers i.e. data, KPIs collection and visualization. Following is the brief explanation of these three layers:

Data Layer:

In architecture this layer is referred as “On Premises Data”. Cloud storage can also be used but because of data privacy issues in Europe, most automotive companies prefer not to store their data on cloud. This layer consists of possible data sets. These data sets can be spread sheets, contractual documentation, data marts, project management tools, etc. The interfaces are drawn on the dataset and not on the KPIs collection layer because these interfaces are related to data sets instead of BI system.

KPI Collection Layer:

The data from interfaces has to go through secured channel, if it is stored on a location which is not available to all of employees. Different companies adopt different authentication schemes e.g. Single Sign-On (SSO), Windows active directory authentication, etc.

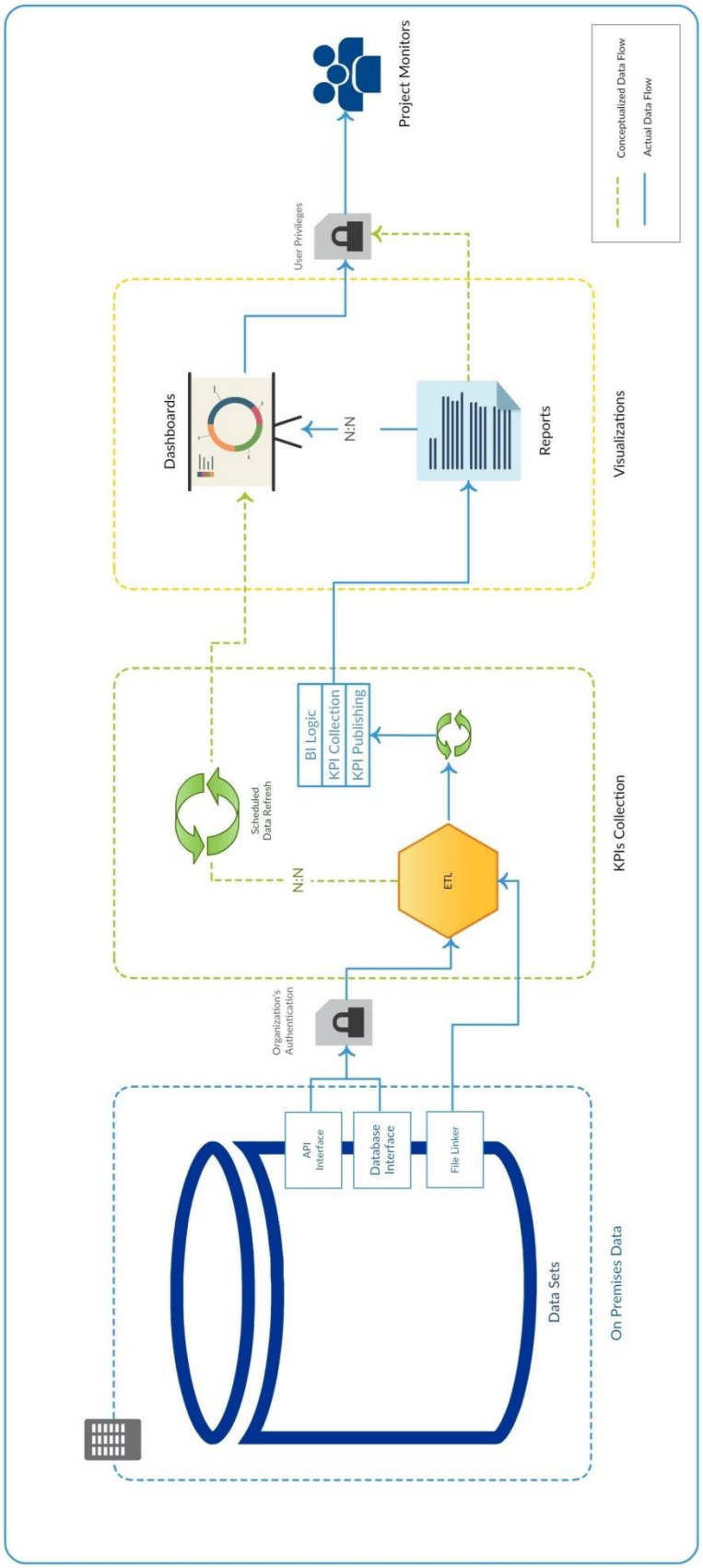


Figure 5.1: Architecture of KPIs collection system

The ETL is responsible to extract data from different data sets using available interfaces. This extracted data is then transformed into format which is readable for the BI system. Usually this transformation is only required for the data which is not in standard tabular form e.g. converting JSON or XML data in to tabular data. The transformed data is then loaded into database used by KPI development tool provided by BI system e.g. QlikView Desktop, Microsoft Power BI Desktop.

The tool used for KPIs collection can be divided further into three layers i.e. logic behind KPIs, KPIs collection and mechanism to publish them for visualizations. The logic behind KPIs is BI logic defined by organization for specific KPIs. A simple example can be of test cases with unknown results, which are derived by subtracting passed and failed test cases from total number of test cases. These logical relationships are then stored as functions or measures depending upon BI system in use. These stored functions are used to collect KPIs from transformed data. Publishing KPIs depends both on collection layer and visualization layer.

Visualization Layer:

Distributing KPIs involves gathering all the data, logics and relationships and then showing them inform of reports or dashboards. Some BI tools work directly on dashboards and whole dashboards are published directly, where other adopt publishing mechanism based on reports. The reports can be completely or partially (visual components of reports) can be published on dashboards.

The project monitors consists of the higher level management, quality managers, product managers, project managers and assessors of standardizations. The dashboards have specific permission level settings thus allowing administrators to grant access to users only for certain dashboards.

The actual data flow in Figure 5.1 is shown with blue lines. The dotted green lines are showing conceptual data flow. This means either it is possible only in some BI systems or it is not a direct link. For example scheduled data refresh for dashboard users seems like they are directly connected to ETL system. Where on every schedule refresh data is first loaded into database of BI development system, goes under all the logical operations and then published onto dashboards via reports. Other than scheduled data refresh developers can perform manual data refresh as demanded by project monitors or after publishing new content onto dashboards. The authentication schemes depend upon type of BI system used and security measures adopted by the organization.

The link between project monitors and reports is also subject to change with different BI solutions. Not all BI solutions follow format of reports and reports maybe replaced with respective visual components. Usually the reports have *N:N* dependency on dashboards

allowing developers to publish dashboards containing information of single or multiple projects. The implementation of this architecture using Microsoft Power BI is discussed in following section.

5.2 Methodology on an Example Project

A state of the art underdevelopment infotainment system developed by a major tier 1 supplier for an OEM is used as an example project here. The automotive business unit team in Mentor Graphics is currently working on the part of this project.

After the evaluation carried out in previous chapter Microsoft Power BI is selected as BI tool to be used for example project. This is because with power BI no licensing commitments are needed for implementation of example projects where QlikView needs licensing for publishing Dashboards on QlikView server. The graphical representation of methodology is presented in Figure 5.2.

Data for tasks, bug fixes, improvements, etc. is stored inform of JIRA issues related to this project. These issues are assigned to specific versions inside JIRA. A milestone is completed when associated version of milestone is released. Some important KPIs here are release dates of upcoming milestones, completion of current milestone (e.g. 70%), open issues in current milestone, etc. JIRA and Microsoft Power BI both supports REST API, which here is used for interfacing. Data related to testing is stored in MySQL data base by testing team. Power BI connects with MySQL using internal MySQL connector. Testing database contains a lot of important KPIs so it more efficient to connect to database instead of connecting to excel test template. Some important KPIs from testing database are total number of test cases, test cases to requirement traceability, passed and failed test cases, etc.

The data related to SCA is stored inside Jenkins build by using FlexeLint on Jenkins web server. Power BI file linker can connect to files stored on local drive, network or web location. Using Power BI file linker SCA summary file from Jenkins server and Technical Requirement Specification (TRS) template from network drive are linked to ETL. TRS is also available in test database but to test the file linking functionality, TRS template file is used. Some important KPIs here are total number of lines of code, number of comments, requirements to test cases traceability, etc. Complete KPIs in example project are presented Appendix C. The authentication for connecting to database used in Mentor Graphics is Mentor Single sign-on (SSO), which is popular authentication method and it is widely used among many medium to large size organizations. Connection with MySQL database may require further authentication as defined by database administrators.

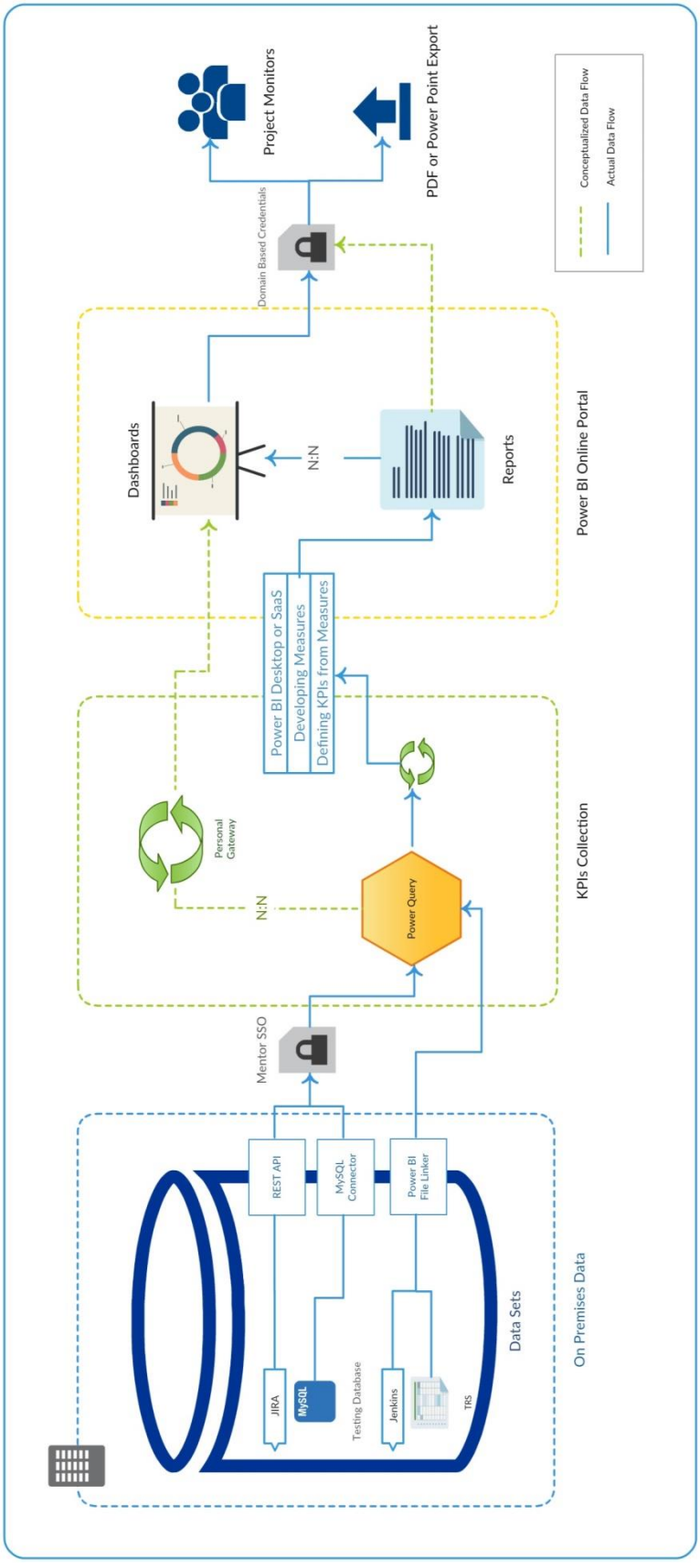


Figure 5.2: Methodology on a project from major tier 1 supplier using Microsoft Power BI

The ETL functionality in Microsoft Power BI is carried out by Power Query. Power Query extracts data from data sets, converts/transforms data and then loads it to database of Power BI Desktop or web application. Microsoft Power BI provides personal gateway for refreshing data without involving desktop or web application. Establishing personal gateway has many security concerns, thus it can be developed as part of future improvements.

KPIs can be defined using Power BI desktop or web application that is why it is overlapping in visualization and KPIs collection layer. Measures inside Power BI can be used to implement logic behind certain KPIs. Using calculated measures, KPIs are developed inform of charts, values or graphs. One report contains several KPIs. Selected reports or KPIs are then published on to dashboards. Power BI dashboards can be shared with interested parties using domain based credentials i.e. employees with same email can use their email to connect to Power BI online portal. The reports and dashboards can also be exported as PDF or power point files.

After publishing all the KPIs inform of reports and publishing required dashboards, an Automotive SPICE assessment was carried out. The actual results of that assessment can't be shared in this research work because of confidentiality issues. The feedback received from assessors showed positive results of the implementation of KPIs collection system on the example project. The part of process monitoring by defining and collecting KPIs is available now using applied methodology. The assessment is discussed in details in Chapter 7. More KPIs are required to fully achieve process performance management. These KPIs are currently not collected because of unavailability of data sets or legacy interfaces of project management tools.

Summary

In this chapter, by considering requirements architecture of new system has been proposed. The architecture is divided into three layer data, logic and visualization. The architecture is proposed with keeping generic structure of BI tools in mind and subject to change with selected BI tool. In second half of the chapter architecture is implemented using Microsoft Power BI on an example project from a major tier 1 supplier. SPICE assessment has shown success of the methodology on example project. Detailed implementation is discussed in next chapter.

Chapter 6

6. Implementation

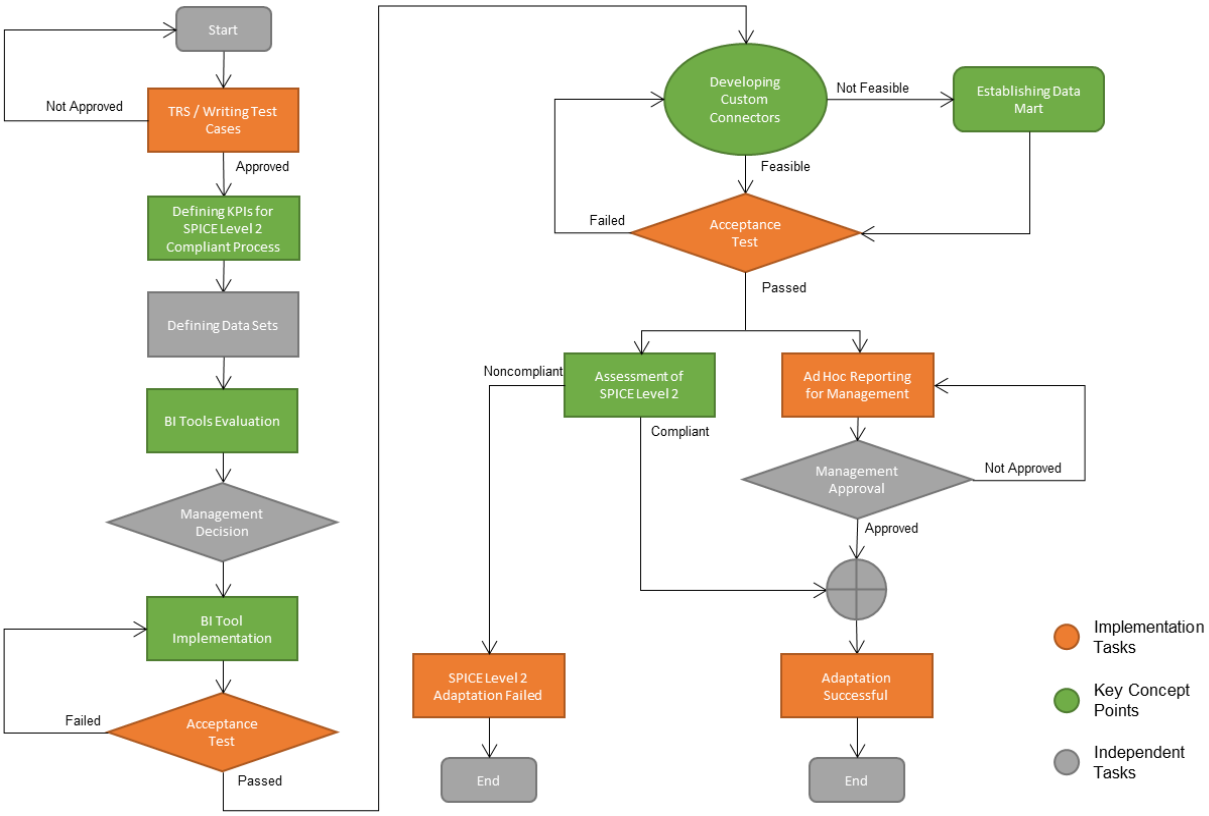


Figure 6.1: Activity diagram for the implementation of the BI System

The implementation of Microsoft Power BI in current environment is discussed in this chapter. Power BI implementation is divided into two major parts, data collection and gathering KPIs out of collected data. These two main parts are further divided into following steps:

Data Collection:

1. Data collection from JIRA using REST API
2. Data collection from test database and Jenkins
3. Transforming collected data

Gathering KPIs:

4. Developing new measures in database
5. Developing reports and publishing dashboards

6.1 Data Collection

Data collection is performed from JIRA, Jenkins and test database. Collecting data from database is really fast and easy to update, but it needs extra resources for maintenance. API calling is more suitable when less overhead is intended and establishing or connecting to a database is not possible.

6.1.1 Data Collection from JIRA using REST API

The REST API offers access to resources via URI paths. To use REST API, applications can make an HTTP request and parse the returned Java Script Object Notation (JSON) response. The methods are the standard HTTP methods like GET, PUT, etc. REST API operates over HTTP or HTTPs making it easy to practice with any programming language or framework. The input and output format for the JIRA REST API is JSON. [26] URIs for JIRA's REST API resource has the following structure:

"http://host:port/context/rest/api-name/api-version/resource-name" [27]

Sending REST requests inside Power BI is possible.

Request:

Following is the sample request to grab everything related to the example project from a major tier 1 supplier:

"http://host:port/rest/api/2/search?jql=project=ProjectKey&maxResults=200"

To create comprehensive KPIs and because of availability of Power Query, minimal filters are used inside above mentioned request. This is good approach to minimize load on JIRA server, and to avoid any inconvenience for other JIRA users. Custom value for "*maxResults"* is used here, because by default, results returned by JIRA REST API are 50.

Response:

The JSON response is quite long, so a sample return of one issue is following:

```
{
   "expand":
```

"schema,names", "startAt": 0, "maxResults": 200,

"total": 49, "issues":

[

{

"expand": "html", "id": "16612",

"self": " http://host:port/rest/api/2/issue/HNTG-155",

"key": "HNTG-155", "fields":

{

"summary": "testing CAN ", "timetracking": null, "issuetype":

{

"self": " http://host:port/rest/api/2/issuetype/5",

"id": "5", "description": "",

"iconUrl": " http://host:port/images/icons/issue_subtask.gif",

"name": "Sub-task", "subtask": true,

…… (More fields)

},…….

In above example, most of field are replaced with standard values because of confidentiality issues. It is difficult to interpret any useful information from above stated JSON response but thanks to Power Query which enables transformation steps on JSON response and then converts it into standard tabular database. To refresh the data, request is sent again, while all the transformation steps are automatically performed on response.

6.1.2 Data Collection from Test Database and Jenkins

A benefit of using database for testing data is to avoid extensive transformations and easy plus fast retrieval of data. In current environment for every project three types of databases are maintained:

- Test plan database
- Test results database

- Technical requirement specification database

TRS database is maintained with test database for the traceability of test cases to requirements. Following sample script is used to connect to database server:

```
let

Source=MySQL.Database("IPAddress","esd_abu_test_results",
[ReturnSingleDatabase=true]),

esd_abu_test_results_majorTier1_project=
Source{[Schema="esd_abu_test_results",Item="majorTier1_project"]}[Data]

in

esd_abu_test_results_majorTier1_project
```

To collect data related to SCA Jenkins' builds are used. REST API can also be used but again there is too much overhead of transformations on JSON response. Using FlexeLint scripting on Jenkins server, SCA information is added into summary file in a fashion that it is easily readable for BI system. Direct connection with summary file provides latest SCA KPIs. The sample request for Jenkins server is following:

```
Source=
Csv.Document(Web.Contents("LocalJenkinsServer/job/MajorTier1_project_master_Flexelint/ws/build/tmp-glibc/deploy/lint/flexelint-summary.txt"),[Delimiter=":", Columns=2, Encoding=1252])
```

Delimiter is use to break long unnecessary strings within a single field.

6.1.3 Transforming Collected Data

The data collected using APIs, database connections or through directly accessing source files is not suitable to gather meaningful KPIs. First the data inform of JSON response need to be converted into tabular form. Power Query supports DAX scripting to convert JSON data into tabular form.

Minor data conversions are carried out on the data retrieved from direct database connection. The FlexeLint files contains too much data, but still script to filter out required data for SCA KPIs is relatively simple compared to JSON conversation scripts.

The complete transformation scripts are provided in Appendix A.

6.2 Gathering KPIs

To gather important KPIs, tabular data is still needed to be modified further. DAX supports creation of new columns called “calculated columns” with dependencies on other tables or columns.

6.2.1 Developing New Measures

A powerful feature of Power BI is concept of “Measures”. Measures are single values which can contain several types of data types e.g. numbers, strings, dates, etc. Following is an example of measure, to calculate release date for next version from data collected through JIRA:

```
Next Version Release Date = FORMAT(MINX(FILTER('Major Tier1 Project',[Fix Version Released]="False" && 'Major Tier1 Project'[Fix Version ReleaseDate]>MINX(FILTER('Major Tier1 Project',[Fix Version Released]="False"),[Fix Version ReleaseDate])),[Fix Version ReleaseDate]), "dd.mm.yyyy")
```

The above code returns single date value for next version release date. The measures used to calculate complex KPIs are much larger and can have dependencies on several columns or tables. Even new data views can be created inside measures and then dependencies can be established with these data views. This approach avoid creation of new tables thus saving calculation time and storage space. Following is an example of measure using data view inside calculation:

```
Duration of Performed Test Cases =
FORMAT(SUMX(SUMMARIZE('esd_abu_test_results majorTier1_project',[TestID],"Time Duration Average",AVERAGE('esd_abu_test_results majorTier1_project'[TestDuration])), [Time Duration Average]), "hh:mm:ss")
```

The “SUMMARIZE” function is usually used to create new tables but here it is just creating a data view inside calculation of a Measure for repeated results average. Measures for important KPIs are given in Appendix B.

6.2.2 Developing Reports and Publishing Dashboard

Power BI provides flexible way of reporting. Single or multiple dashboards can be created for a single project. Each dashboard is fed by the reports which are created using transformed data and measures. A complete report or selected components of reports can be then deployed to dashboard.

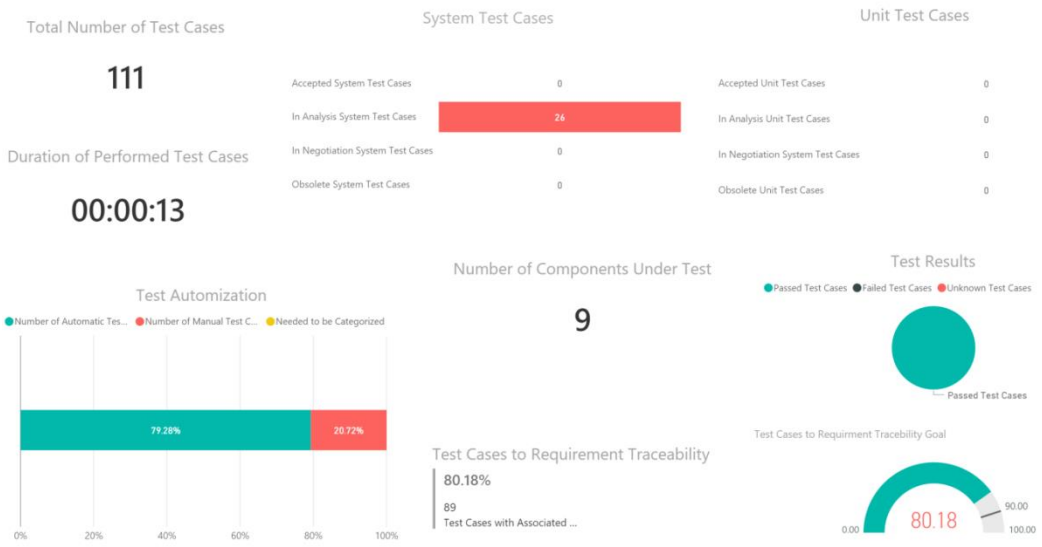


Figure 6.2: Report with KPIs from testing and SCA

The exclusive type of reporting is really useful. For example in Figure 6.2 higher management is only interested in requirement traceability with test cases, and have no interest in total number of passed test cases, so the KPIs related to traceability is only pinned to executive dashboard, where total number of test cases is important for testing manager, so KPI for passed test cases can be exclusively pinned on dashboard for test manager. Reports and dashboards are attached in Appendix C. The possible project monitors are defined in Section (3.3). These monitors are responsible for different aspects of a project. One or more monitors are interested in exclusive dashboards. From perspective of process performance planning, project monitors are looking for some KPIs which may not interest other project monitors. Following table shows relation between some important KPIs, project monitors and proposed dashboards:

Project Monitors	Related KPIs	Proposed Dashboard
Process and Quality Manager	Should have access to all KPIs	Should have access to all Dashboards
Product Delivery Manager	Required, planned and accepted working hours Milestone release dates and competition Etc.	Dashboard showing deadlines and resource allocation in several projects
Test Manager	Requirements to test cases traceability SCA information Passed, failed test cases Test automation status Etc.	Dashboards showing critical testing KPIs from several projects
Project Manager	All KPIs related to his/hers projects	Dashboard showing critical KPIs from his/hers projects
Higher Level Management	Resource allocation KPIs KPIs related to deadlines and delivery Projects' overall completion time Etc.	Dashboard showing completion of projects, allocated resources to projects, delayed projects, completion time of recently completed projects

Table 6.1: Related KPIs and proposed dashboards for project monitors

Summary

The evaluation of proposed approached carried out in Chapter 4 led to the implementation of Microsoft Power BI. Power BI provides out of the box interfaces with famous type of data sources and services.

Using APIs to collect data results, causes less overhead by saving resources and load on data source but a lot of data transformation is need on data collected using API requests. Here data collected from JIRA result in JSON which needs fair amount of transformation before being able to deliver any meaningful KPIs. Establishing database as a proxy seems easy and faster but it needs a lot of effort and handsome amount of resources for maintenance of database.

Concept of Measures is really useful and powerful in Power BI. DAX is used to calculate important Measures which help in assembling important KPIs. Reports contain one or more KPIs, which can be shared either exclusive to specific users or publically with many users who are using dashboards.

Chapter 7

7. Assessment and Evaluation

This chapter discusses assessment of Automotive SPICE and evaluation of implementation by evaluating research questions presented in Section (1.2). First half of chapter discusses about SPICE assessment which is carried out after implementation of BI system and effects of BI system on achieving Automotive SPICE level 2. The second half discusses implemented tool as a monitoring system for managers and evaluation of achievements of this thesis.

7.1 Automotive SPICE Assessment

Automotive SPICE reference model, process dimension, capability dimension, practices and resources are discussed in Section (2.1). After implementation of KPI monitoring system, an internal Automotive SPICE assessment is carried out by designated assessors in current environment.

The actual Automotive SPICE assessment, which is carried out in Mentor Graphics can neither be disclosed nor be discussed here because of confidentiality issues. According to Automotive SPICE process assessment model:

"As a fundamental rule, assessment results and the knowledge Obtained in the course of an assessment must be treated as confidential by all persons and organisations involved. Assessments covering several levels of suppliers require a separate agreement. The Accessing Organisation (AO) is the owner of the assessment results. The forwarding of assessment results to third parties should be agreed in writing"
[3, Page 33]

The purpose of KPIs collection is to help with achievement of Automotive SPICE level 2. The process performance management attribute which has already been explained in Section (2.2) is affected directly after the implementation of KPIs collection system. The mapping of process performance management attribute onto possible current development approach after the implementation of KPIs collection system is given in following Table:

PA 2.1 References	**Situation after the implementation KPIs collection system**
Achievements	The achievements after implementation are: a) Objective for the performance is identified by the respective team e.g. 90% traceability between test cases and requirements b) The process performance planning is team leader's job,

	KPIs are goals related to performance planning. KPIs are already defined by process improvement manager c) KPIs collection System is used to monitor performance of the process by collecting and visualizing KPIs related to process d) Performance adjustment is job of responsible team, e.g. developers focus more on debugging to reduce bugs per lines of code e) Responsible managers define responsibilities and authorities e.g. product delivery manager should take action on missing milestones delivery by asking the respective projective manager who in turn will identify the problem and take further actions f) Employees should have sufficient skills to carry out advised improvements in process g) Some processes need certain resources e.g. automated testing needs tools for test automation and required knowledge for testing engineers h) Already latest and reliable communication channels are available
Generic Practices	**GP 2.1.3 The performance of process is monitored by collecting and visualizing KPIs related to process using KPIs collection system** [ACHIEVEMENT c]
Generic resources	**Licencing for Microsoft Power BI required for implementing it on wide scale. At the start of the project data interfaces need to be established with ETL system. Also a developer is required for reports and dashboard development and management.**

Table 7.1: Mapping of PA 2.1 onto KPIs collection system

Only GP 2.1.3 is discussed here as other generic practices are not directly related to the KPIs collection system. GP 2.1.3 is directly related to ACHIEVEMENT C which deals with monitoring the performance of process. The acceptance for publishing actual assessment results for Mentor Graphics is not granted. Nevertheless following are some key points provided by assessors regarding implementation of evaluated KPIs collection system:

- Monitoring for work estimation needed to be more thorough e.g. planned working hours, actual working hours
- The process performance attribute is largely achieved
- Code coverage info is only applicable for unit tests and can only be measured with a valid test harness, e.g. IBM RT/RT tool, VectorCast, etc.

- For Projects from OEMs like BMW or Daimler, KPIs should also measure code coverage. It is not done due to a lack of time of the developers and testers.
- Code coverage KPIs are mediatory for Automotive SPICE level 2 compliance
- Integration and system testing should provide KPIs related to function coverage and call coverage, which is also not measured yet in testing team.

The implementation of evaluated BI tool has played its role in measuring performance of several processes. This approach can be used in future to other projects in help achieving Automotive SPICE level 2.

7.2 Evaluation of Research Questions

The higher management is satisfied with the tool availability. Before being used into actual production environment it is needed to be implemented on to all other projects which grab interest of higher management.

The methods used in this thesis can now address the research questions as described in Section (1.2). An evaluation of these questions is as follows:

Can BI tool help in achievement of Automotive SPICE Level 2?

As shown in Section (6.2.2) using a BI tool, important KPIs can be automatically collected from different processes. The process performance management attribute can be satisfied with help of KPIs collection. So it is possible to monitor performance of processes using a BI tool.

Commercial or open source solution?

As discussed in Section (4.4). The adoption of commercial or adapting an open source solution depends upon skill set of developers, organisation's long term goals and requirement within current environment. Commercial solutions seem expensive but they break even or sometimes proven to be beneficial as compared to open source solutions in the course of time.

Which approach is more efficient for an automotive supplier i.e. either data collection through data marts or API calling to gather data from data sets?

As discussed in Section (6.1) with APIs, data collection may results in less overhead but more transformation steps may require. Establishing database is faster but with causes more overhead and also not possible in case of every software development tool. Thus the APIs are preferred to collect data.

Impact of solution on current processes?

It has been proved that BI solution can be used in current environment. More data sources are needed to gather missing KPIs. Some extra effort is also required to make data sources more reliable for effective decision making and consistent KPIs.

Chapter 8

8. Summary and Outlook

Using BI tools for achieving compliances is a fresh technique. The monitoring inside automotive domain specific ALMs focuses more on quality standards and less on a comprehensive reporting system. The monitoring approach using BI tool is highly comprehensive, scalable and is not bound only to processes which are directly related to development. The BI tool can be extended for reporting into other areas of organisation like finance and human resources. Using a common reporting system can also avoid confusions and help standardising reporting structure within an organisation.

8.1 Future Work

The adapted tool i.e. MS Power BI is relatively young tool within BI tools industry. New features are coming in Power BI constantly. More and more tools are giving easy connectivity with BI tools. So in future less effort will be required for interfacing. The roadmap for Power BI from Microsoft contains on premises reporting using SQL Server 2016, improved layouts for mobile devices, more interactive reports, and preserving the structure of BI system both on cloud and on premises.

This thesis is rather evaluation of implementation of BI tool for an automotive supplier rather providing complete solution throughout the organisation. Automotive SPICE is not directly related to organisation and related to individual projects and processes, so implementation of BI tool needed to be expanded towards other projects.

There are some tools which support legacy APIs e.g. OpenAir. Developing interfaces with legacy tools can be carried out in future. There is great deal of future work in implementation of BI tools within automotive suppliers which can be carried out using this thesis as a reference point.

8.2 Summary

Automotive SPICE in automotive Industry is becoming more and more significant. It addresses best practices, tracking and monitoring of processes during software development. Automotive manufacturers are facing challenges like increased reliability requirements with increase in complexity, short time periods for product roll out, using legal components and assuring the interoperability of components from different suppliers, because of these challenges many automotive manufacturers are demanding Automotive SPICE level 2 from their suppliers.

As tracking and monitoring is big part of level 2 compliance, it is recommended to capture KPIs for projects. A KPIs collection system should have good graphical representation capabilities with scalability and flexibility. System should also be able to work within secure environment while still be able to communicate with external resources. Different managers have different requirements for KPIs. System is intended mainly to help achieving level 2 Automotive SPICE compliance and to give critical information to higher level management.

Implementation phase requires good knowledge of software development norms, practices and tools used by automotive suppliers. Understanding of datasets and their interfacing capabilities is also vital. With different tools used among different suppliers presenting a standard solution for tracking and monitoring of process performance is almost implausible. Automotive SPICE can be used both by OEMs and automotive suppliers. Achieving level 2 Automotive SPICE compliance is quite challenging and implementing a BI system as a monitoring system is quite new idea for automotive industry. Failure is usually caused in implementation schemes due to absence of knowledge about possible approaches, lacking technical expertise and fear of big investment and little gain from higher management.

Developing BI tool from scratch is out of the scope for this thesis because of time constraints and limited resources. Open source widget based web solution is cost effective but less interactive and provides short time solution. Selecting API calling, establishing data mart or both for interfacing depends upon selected BI tool. QlikView is recommended for immediate on premises implementation. It has SQL like scripting but still extra training is needed for developers. Power BI is more user friendly, cost effective, suites skill set of most of people and has very good roadmap for integration with other Microsoft reporting solutions

The evaluation of approaches in Chapter 4 led to the implementation of Microsoft Power BI. Power BI provides out of the box interfaces with famous type of data sources and services. Using APIs to collect data results in less overhead by saving resources and load on data source, but a lot of data transformation is need on the data collected using API requests. Here data collected from JIRA result in JSON which needs fair amount of transformation before being able to deliver any meaningful KPIs. Establishing database as a proxy seems easy and faster but it needs a lot of effort and handsome amount of resources for maintenance of database.

Concept of Measures is really useful and powerful in Power BI. DAX is used to calculate important Measures which helps in assembling important KPIs. Reports contain one or more KPIs, which can be shared either exclusive to specific users or publically for many users using dashboards. The implementation of BI system for monitoring is as correct as data sources used within project.

Bibliography

[1] Klaus Hoermann, Lars Dittmann, Joerg Zimmer Markus Mueller, *Automotive SPICE in Practice: Surviving Interpretation and Assessment*.: O'Reilly Media, Inc., 2008.

[2] Axel Hoffmann, Jan Marco Leimeister, Marina Berkovich, Helmut Krcmar Ruth Klendauer, "Using the IDEAL Software Process Improvement Model for the Implementation of Automotive SPICE," Zurich, Reserach Paper 2012.

[3] Henrich Druck, "Autmotive SPICE Process Assessment Model: 1st edition," VDA QMC, Frankfurt, 2010.

[4] Impronova AB, Lindome Alec Dorling, "The Role of Process Standards in Automotive Systems Development," Research Paper ISBN: 978-0-7695-3262-2, 2008.

[5] VDA QMC Working Group 13, Automotive SIG, "Automotive SPICE Process Assessment / Reference Model Version 3.0," Refrence Model 2015.

[6] Suneel Sabar, "Software Process Improvement and," Linköpings universitet, Linköping, Sweeden, Master Thesis 2011.

[7] The Hansen Report on Automotive Electronics, "Hansen Monthly Report," Report 2005.

[8] Introduction to KPIs. [Online]. http://keyituk.com/wordpress/wp-content/uploads/2013/04/kpi.png

[9] David Parmenter, *Key Performance Indicators: Developing, Implementing, and Using Winning KPIs*.: ohn Wiley & Sons, 2015.

[10] Christian Brecher, *Integrative Production Technology for High-Wage Countries*.: Springer Science & Business Media, 2011.

[11] K.R. Radhika Prathima, Fayaz Nirmala S. Guptha, "Comparative Analysis of Online Analytical Processing and Reporting Tools," Reva ITM, Banalore, Research Paper 2010.

[12] Tableau Customers. [Online]. https://www.tableau.com/about/customers

[13] Microsoft Customers. [Online]. https://customers.microsoft.com/Pages/CustomerStory.aspx?recid=22883

[14] Polarion Software America & Asia, "Küster Automotive Success Story," online available at https://www.polarion.com/hubfs/Docs/customer-success-stories/Kuester-Automotive-

Customer-Success-Story.pdf?t=1466108608900.

[15] Intland Softwere, "Continental Customer Success Story," 2016, Online available at http://intland.com/wp-content/uploads/2014/02/success-story-continental.pdf.

[16] About JIRA ORIGINS. [Online]. https://confluence.atlassian.com/pages/viewpage.action?pageId=223219957

[17] NetSuite, "NetSuite OpenAir: The World's #1 Cloud Professional Service Autmation Solution," *product brochure*, Online available at http://www.netsuite.com/portal/common/pdf/ns-datasheet-openair.pdf.

[18] Rainer Anders, "Empowering organisations with SPICE Software process improvement the practioner's way," Software Quality Systems, White Paper.

[19] M. SCHWARZ, E. UGLJESA, P. HOLUB, A. HAYEK J. BOERCSOEK, "High-Availability Controller Concept for Steering Systems," Computer Architecture and System Programming, Kassel, Research Paper ISBN: 978-1-61804-056-5 ,.

[20] Sarah Moser, "Benefits of Automotive SPICE," Mentor Graphics, Concept Presentation 2013.

[21] Sarah Moser, "Propsal of KPI collection Systems," Mentor Graphics, Villingen-Schwenningen, Proposals Presentation 2013.

[22] Polarion Software, "Accelerate Innovation with Unified Application Lifecycle Managment," White Paper 2016.

[23] Part of figures for roles. [Online]. http://ayandy.getsimplesite.com/projectmanager.png, https://cdn4.iconfinder.com/data/icons/free-large-boss-icon-set/512/Engineer.png, http://icons.iconarchive.com/icons/visualpharm/office-space/128/user-icon.png

[24] Qlik, "QlikView Development and Deployment Architecture," Technical Brief 2011.

[25] Alberto Ferrari, Ariel Netz, Carl Perry, Cindy Song, Faisal Mohamood Adam Wilson, "Bring your data to life with Power BI," Microsoft, White Paper 2015.

[26] About JIRA REST APIs. [Online]. https://developer.atlassian.com/jiradev/jira-apis/jira-rest-apis

[27] JIRA REST API Refrence. [Online]. https://docs.atlassian.com/jira/REST/latest/

[28] HP Luhn, "A Business Intelligence System," *IBM Journal*, 1958.

Appendix A

JSON data into tabular form conversion

```
let

    Source =
Json.Document(Web.Contents("http://jira.alm.mentorg.com:8080/rest/api/2/search?jql=project=HNTG&maxResults=200")),

    issues = Source[issues],

    #"Converted to Table" = Table.FromList(issues, Splitter.SplitByNothing(), null, null,
ExtraValues.Error),

    #"Expanded Column1" = Table.ExpandRecordColumn(#"Converted to Table", "Column1",
{"expand", "id", "self", "key", "fields"}, {"Column1.expand", "Column1.id", "Column1.self",
"Column1.key", "Column1.fields"}),

    #"Removed Columns" = Table.RemoveColumns(#"Expanded
Column1",{"Column1.expand"}),

    #"Expanded Column1.fields1" = Table.ExpandRecordColumn(#"Removed Columns",
"Column1.fields", {"customfield_14150", "customfield_14154", "customfield_11040",
"customfield_14151", "customfield_14152", "customfield_14157", "customfield_11440",
"fixVersions", "customfield_14156", "customfield_13346", "customfield_13742",
"customfield_13345", "resolution", "customfield_13741", "customfield_13347",
"customfield_13743", "lastViewed", "customfield_14146", "customfield_14147",
"customfield_14145", "priority", "customfield_10740", "labels", "customfield_14149",
"aggregatetimeoriginalestimate", "timeestimate", "versions", "issuelinks", "assignee",
"status", "components", "customfield_13441", "customfield_13440", "customfield_13841",
"customfield_13840", "customfield_13843", "customfield_13842", "aggregatetimeestimate",
"creator", "customfield_14241", "subtasks", "customfield_10040", "reporter",
"aggregateprogress", "customfield_10444", "customfield_11645", "customfield_13944",
"customfield_11644", "customfield_13943", "customfield_11647", "customfield_11646",
"customfield_13945", "progress", "votes", "issuetype", "timespent", "project",
"customfield_13940", "aggregatetimespent", "customfield_11643", "customfield_13942",
"customfield_11642", "customfield_10942", "customfield_12846", "resolutiondate",
"workratio", "customfield_14183", "watches", "customfield_14180", "customfield_14340",
"customfield_14341", "created", "customfield_10020", "customfield_14342",
```

"customfield_14189", "customfield_10024", "customfield_11743", "updated",
"customfield_14172", "customfield_14054", "customfield_14175", "customfield_14055",
"customfield_14052", "customfield_14173", "customfield_14053", "customfield_14174",
"timeoriginalestimate", "customfield_14179", "description", "customfield_10010",
"customfield_14177", "customfield_14056", "customfield_10011", "customfield_14178",
"customfield_11742", "customfield_12949", "summary", "customfield_14160",
"customfield_14161", "customfield_14043", "customfield_14164", "customfield_14044",
"customfield_14165", "customfield_14163", "customfield_14168", "customfield_10240",
"customfield_14048", "customfield_14169", "customfield_14045", "customfield_14166",
"customfield_13751", "customfield_14046", "customfield_13750", "customfield_14167",
"customfield_13753", "customfield_10640", "customfield_14049", "customfield_13746",
"environment", "customfield_13748", "customfield_13747", "customfield_13749", "duedate",
"parent"}, {"Column1.fields.customfield_14150", "Column1.fields.customfield_14154",
"Column1.fields.customfield_11040", "Column1.fields.customfield_14151",
"Column1.fields.customfield_14152", "Column1.fields.customfield_14157",
"Column1.fields.customfield_11440", "Column1.fields.fixVersions",
"Column1.fields.customfield_14156", "Column1.fields.customfield_13346",
"Column1.fields.customfield_13742", "Column1.fields.customfield_13345",
"Column1.fields.resolution", "Column1.fields.customfield_13741",
"Column1.fields.customfield_13347", "Column1.fields.customfield_13743",
"Column1.fields.lastViewed", "Column1.fields.customfield_14146",
"Column1.fields.customfield_14147", "Column1.fields.customfield_14145",
"Column1.fields.priority", "Column1.fields.customfield_10740", "Column1.fields.labels",
"Column1.fields.customfield_14149", "Column1.fields.aggregatetimeoriginalestimate",
"Column1.fields.timeestimate", "Column1.fields.versions", "Column1.fields.issuelinks",
"Column1.fields.assignee", "Column1.fields.status", "Column1.fields.components",
"Column1.fields.customfield_13441", "Column1.fields.customfield_13440",
"Column1.fields.customfield_13841", "Column1.fields.customfield_13840",
"Column1.fields.customfield_13843", "Column1.fields.customfield_13842",
"Column1.fields.aggregatetimeestimate", "Column1.fields.creator",
"Column1.fields.customfield_14241", "Column1.fields.subtasks",
"Column1.fields.customfield_10040", "Column1.fields.reporter",
"Column1.fields.aggregateprogress", "Column1.fields.customfield_10444",
"Column1.fields.customfield_11645", "Column1.fields.customfield_13944",
"Column1.fields.customfield_11644", "Column1.fields.customfield_13943",
"Column1.fields.customfield_11647", "Column1.fields.customfield_11646",
"Column1.fields.customfield_13945", "Column1.fields.progress", "Column1.fields.votes",
"Column1.fields.issuetype", "Column1.fields.timespent", "Column1.fields.project",
"Column1.fields.customfield_13940", "Column1.fields.aggregatetimespent",
"Column1.fields.customfield_11643", "Column1.fields.customfield_13942",

"Column1.fields.customfield_11642", "Column1.fields.customfield_10942",
"Column1.fields.customfield_12846", "Column1.fields.resolutiondate",
"Column1.fields.workratio", "Column1.fields.customfield_14183",
"Column1.fields.watches", "Column1.fields.customfield_14180",
"Column1.fields.customfield_14340", "Column1.fields.customfield_14341",
"Column1.fields.created", "Column1.fields.customfield_10020",
"Column1.fields.customfield_14342", "Column1.fields.customfield_14189",
"Column1.fields.customfield_10024", "Column1.fields.customfield_11743",
"Column1.fields.updated", "Column1.fields.customfield_14172",
"Column1.fields.customfield_14054", "Column1.fields.customfield_14175",
"Column1.fields.customfield_14055", "Column1.fields.customfield_14052",
"Column1.fields.customfield_14173", "Column1.fields.customfield_14053",
"Column1.fields.customfield_14174", "Column1.fields.timeoriginalestimate",
"Column1.fields.customfield_14179", "Column1.fields.description",
"Column1.fields.customfield_10010", "Column1.fields.customfield_14177",
"Column1.fields.customfield_14056", "Column1.fields.customfield_10011",
"Column1.fields.customfield_14178", "Column1.fields.customfield_11742",
"Column1.fields.customfield_12949", "Column1.fields.summary",
"Column1.fields.customfield_14160", "Column1.fields.customfield_14161",
"Column1.fields.customfield_14043", "Column1.fields.customfield_14164",
"Column1.fields.customfield_14044", "Column1.fields.customfield_14165",
"Column1.fields.customfield_14163", "Column1.fields.customfield_14168",
"Column1.fields.customfield_10240", "Column1.fields.customfield_14048",
"Column1.fields.customfield_14169", "Column1.fields.customfield_14045",
"Column1.fields.customfield_14166", "Column1.fields.customfield_13751",
"Column1.fields.customfield_14046", "Column1.fields.customfield_13750",
"Column1.fields.customfield_14167", "Column1.fields.customfield_13753",
"Column1.fields.customfield_10640", "Column1.fields.customfield_14049",
"Column1.fields.customfield_13746", "Column1.fields.environment",
"Column1.fields.customfield_13748", "Column1.fields.customfield_13747",
"Column1.fields.customfield_13749", "Column1.fields.duedate", "Column1.fields.parent"}),

#"Removed Columns1" = Table.RemoveColumns(#"Expanded
Column1.fields1",{"Column1.fields.customfield_14150",
"Column1.fields.customfield_14154", "Column1.fields.customfield_11040",
"Column1.fields.customfield_14151", "Column1.fields.customfield_14152",
"Column1.fields.customfield_14157", "Column1.fields.customfield_11440"}),

#"Expanded Column1.fields.fixVersions" = Table.ExpandListColumn(#"Removed
Columns1", "Column1.fields.fixVersions"),

```
    #"Expanded Column1.fields.fixVersions1" = Table.ExpandRecordColumn(#"Expanded
Column1.fields.fixVersions", "Column1.fields.fixVersions", {"self", "id", "description",
"name", "archived", "released", "releaseDate"}, {"Column1.fields.fixVersions.self",
"Column1.fields.fixVersions.id", "Column1.fields.fixVersions.description",
"Column1.fields.fixVersions.name", "Column1.fields.fixVersions.archived",
"Column1.fields.fixVersions.released", "Column1.fields.fixVersions.releaseDate"}),

    #"Removed Columns2" = Table.RemoveColumns(#"Expanded
Column1.fields.fixVersions1",{"Column1.fields.fixVersions.self",
"Column1.fields.customfield_14156", "Column1.fields.customfield_13346",
"Column1.fields.customfield_13742", "Column1.fields.customfield_13345"}),

    #"Expanded Column1.fields.resolution" = Table.ExpandRecordColumn(#"Removed
Columns2", "Column1.fields.resolution", {"self", "id", "description", "name"},
{"Column1.fields.resolution.self", "Column1.fields.resolution.id",
"Column1.fields.resolution.description", "Column1.fields.resolution.name"}),

    #"Removed Columns3" = Table.RemoveColumns(#"Expanded
Column1.fields.resolution",{"Column1.fields.resolution.self",
"Column1.fields.customfield_13741", "Column1.fields.customfield_13347",
"Column1.fields.customfield_13743", "Column1.fields.lastViewed",
"Column1.fields.customfield_14146", "Column1.fields.customfield_14147",
"Column1.fields.customfield_14145"}),

    #"Expanded Column1.fields.priority" = Table.ExpandRecordColumn(#"Removed
Columns3", "Column1.fields.priority", {"self", "iconUrl", "name", "id"},
{"Column1.fields.priority.self", "Column1.fields.priority.iconUrl",
"Column1.fields.priority.name", "Column1.fields.priority.id"}),

    #"Removed Columns4" = Table.RemoveColumns(#"Expanded
Column1.fields.priority",{"Column1.fields.priority.self", "Column1.fields.priority.iconUrl",
"Column1.fields.customfield_10740"}),

    #"Expanded Column1.fields.labels" = Table.ExpandListColumn(#"Removed Columns4",
"Column1.fields.labels"),

    #"Removed Columns5" = Table.RemoveColumns(#"Expanded
Column1.fields.labels",{"Column1.fields.labels", "Column1.fields.customfield_14149",
"Column1.fields.aggregatetimeoriginalestimate", "Column1.fields.timeestimate"}),

    #"Expanded Column1.fields.versions" = Table.ExpandListColumn(#"Removed
Columns5", "Column1.fields.versions"),
```

```
    #"Expanded Column1.fields.versions1" = Table.ExpandRecordColumn(#"Expanded
Column1.fields.versions", "Column1.fields.versions", {"self", "id", "description", "name",
"archived", "released", "releaseDate"}, {"Column1.fields.versions.self",
"Column1.fields.versions.id", "Column1.fields.versions.description",
"Column1.fields.versions.name", "Column1.fields.versions.archived",
"Column1.fields.versions.released", "Column1.fields.versions.releaseDate"}),

    #"Removed Columns6" = Table.RemoveColumns(#"Expanded
Column1.fields.versions1",{"Column1.fields.versions.self"}),

    #"Expanded Column1.fields.issuelinks" = Table.ExpandListColumn(#"Removed
Columns6", "Column1.fields.issuelinks"),

    #"Expanded Column1.fields.issuelinks1" = Table.ExpandRecordColumn(#"Expanded
Column1.fields.issuelinks", "Column1.fields.issuelinks", {"id", "self", "type", "inwardIssue",
"outwardIssue"}, {"Column1.fields.issuelinks.id", "Column1.fields.issuelinks.self",
"Column1.fields.issuelinks.type", "Column1.fields.issuelinks.inwardIssue",
"Column1.fields.issuelinks.outwardIssue"}),

    #"Removed Columns7" = Table.RemoveColumns(#"Expanded
Column1.fields.issuelinks1",{"Column1.fields.issuelinks.id",
"Column1.fields.issuelinks.self", "Column1.fields.issuelinks.type",
"Column1.fields.issuelinks.inwardIssue", "Column1.fields.issuelinks.outwardIssue"}),

    #"Expanded Column1.fields.assignee" = Table.ExpandRecordColumn(#"Removed
Columns7", "Column1.fields.assignee", {"displayName"},
{"Column1.fields.assignee.displayName"}),

    #"Expanded Column1.fields.status" = Table.ExpandRecordColumn(#"Expanded
Column1.fields.assignee", "Column1.fields.status", {"self", "description", "iconUrl", "name",
"id", "statusCategory"}, {"Column1.fields.status.self", "Column1.fields.status.description",
"Column1.fields.status.iconUrl", "Column1.fields.status.name", "Column1.fields.status.id",
"Column1.fields.status.statusCategory"}),

    #"Removed Columns8" = Table.RemoveColumns(#"Expanded
Column1.fields.status",{"Column1.fields.status.self", "Column1.fields.status.iconUrl"}),

    #"Expanded Column1.fields.status.statusCategory" =
Table.ExpandRecordColumn(#"Removed Columns8",
"Column1.fields.status.statusCategory", {"self", "id", "key", "colorName", "name"},
{"Column1.fields.status.statusCategory.self", "Column1.fields.status.statusCategory.id",
"Column1.fields.status.statusCategory.key",
```

```
"Column1.fields.status.statusCategory.colorName",
"Column1.fields.status.statusCategory.name"}),

    #"Removed Columns9" = Table.RemoveColumns(#"Expanded
Column1.fields.status.statusCategory",{"Column1.fields.status.statusCategory.self"}),

    #"Expanded Column1.fields.components" = Table.ExpandListColumn(#"Removed
Columns9", "Column1.fields.components"),

    #"Expanded Column1.fields.components1" = Table.ExpandRecordColumn(#"Expanded
Column1.fields.components", "Column1.fields.components", {"self", "id", "name",
"description"}, {"Column1.fields.components.self", "Column1.fields.components.id",
"Column1.fields.components.name", "Column1.fields.components.description"}),

    #"Removed Columns10" = Table.RemoveColumns(#"Expanded
Column1.fields.components1",{"Column1.fields.components.self",
"Column1.fields.components.id", "Column1.fields.components.name",
"Column1.fields.components.description", "Column1.fields.customfield_13441",
"Column1.fields.customfield_13440", "Column1.fields.customfield_13841",
"Column1.fields.customfield_13840", "Column1.fields.customfield_13843",
"Column1.fields.customfield_13842", "Column1.fields.aggregatetimeestimate"}),

    #"Expanded Column1.fields.creator" = Table.ExpandRecordColumn(#"Removed
Columns10", "Column1.fields.creator", {"displayName"},
{"Column1.fields.creator.displayName"}),

    #"Removed Columns11" = Table.RemoveColumns(#"Expanded
Column1.fields.creator",{"Column1.fields.customfield_14241"}),

    #"Expanded Column1.fields.subtasks" = Table.ExpandListColumn(#"Removed
Columns11", "Column1.fields.subtasks"),

    #"Removed Columns12" = Table.RemoveColumns(#"Expanded
Column1.fields.subtasks",{"Column1.fields.subtasks",
"Column1.fields.customfield_10040"}),

    #"Expanded Column1.fields.reporter" = Table.ExpandRecordColumn(#"Removed
Columns12", "Column1.fields.reporter", {"displayName"},
{"Column1.fields.reporter.displayName"}),

    #"Expanded Column1.fields.aggregateprogress" =
Table.ExpandRecordColumn(#"Expanded Column1.fields.reporter",
"Column1.fields.aggregateprogress", {"progress", "total", "percent"},
```

```
{"Column1.fields.aggregateprogress.progress", "Column1.fields.aggregateprogress.total",
"Column1.fields.aggregateprogress.percent"}),

    #"Removed Columns13" = Table.RemoveColumns(#"Expanded
Column1.fields.aggregateprogress",{"Column1.fields.customfield_10444",
"Column1.fields.customfield_11645", "Column1.fields.customfield_13944",
"Column1.fields.customfield_11644", "Column1.fields.customfield_13943",
"Column1.fields.customfield_11647", "Column1.fields.customfield_11646",
"Column1.fields.customfield_13945"}),

    #"Expanded Column1.fields.progress" = Table.ExpandRecordColumn(#"Removed
Columns13", "Column1.fields.progress", {"progress", "total", "percent"},
{"Column1.fields.progress.progress", "Column1.fields.progress.total",
"Column1.fields.progress.percent"}),

    #"Removed Columns14" = Table.RemoveColumns(#"Expanded
Column1.fields.progress",{"Column1.fields.progress.progress",
"Column1.fields.progress.total", "Column1.fields.progress.percent",
"Column1.fields.votes"}),

    #"Expanded Column1.fields.issuetype" = Table.ExpandRecordColumn(#"Removed
Columns14", "Column1.fields.issuetype", {"self", "id", "description", "iconUrl", "name",
"subtask"}, {"Column1.fields.issuetype.self", "Column1.fields.issuetype.id",
"Column1.fields.issuetype.description", "Column1.fields.issuetype.iconUrl",
"Column1.fields.issuetype.name", "Column1.fields.issuetype.subtask"}),

    #"Removed Columns15" = Table.RemoveColumns(#"Expanded
Column1.fields.issuetype",{"Column1.fields.issuetype.self",
"Column1.fields.issuetype.iconUrl", "Column1.fields.issuetype.subtask",
"Column1.fields.timespent"}),

    #"Expanded Column1.fields.project" = Table.ExpandRecordColumn(#"Removed
Columns15", "Column1.fields.project", {"self", "id", "key", "name", "avatarUrls",
"projectCategory"}, {"Column1.fields.project.self", "Column1.fields.project.id",
"Column1.fields.project.key", "Column1.fields.project.name",
"Column1.fields.project.avatarUrls", "Column1.fields.project.projectCategory"}),

    #"Removed Columns16" = Table.RemoveColumns(#"Expanded
Column1.fields.project",{"Column1.fields.project.self",
"Column1.fields.project.avatarUrls"}),

    #"Expanded Column1.fields.project.projectCategory" =
Table.ExpandRecordColumn(#"Removed Columns16",
```

```
"Column1.fields.project.projectCategory", {"self", "id", "description", "name"},
{"Column1.fields.project.projectCategory.self", "Column1.fields.project.projectCategory.id",
"Column1.fields.project.projectCategory.description",
"Column1.fields.project.projectCategory.name"}),

    #"Removed Columns17" = Table.RemoveColumns(#"Expanded
Column1.fields.project.projectCategory",{"Column1.fields.project.projectCategory.self",
"Column1.fields.customfield_13940", "Column1.fields.aggregatetimespent", "Column1.id",
"Column1.fields.customfield_13942", "Column1.fields.customfield_10942",
"Column1.fields.customfield_12846", "Column1.fields.customfield_14183"}),

    #"Expanded Column1.fields.watches" = Table.ExpandRecordColumn(#"Removed
Columns17", "Column1.fields.watches", {"watchCount"},
{"Column1.fields.watches.watchCount"}),

    #"Removed Columns18" = Table.RemoveColumns(#"Expanded
Column1.fields.watches",{"Column1.fields.customfield_14180",
"Column1.fields.customfield_14340", "Column1.fields.customfield_14341",
"Column1.fields.customfield_10020", "Column1.fields.customfield_14342",
"Column1.fields.customfield_14189", "Column1.fields.customfield_10024",
"Column1.fields.customfield_11743", "Column1.fields.customfield_14172",
"Column1.fields.customfield_14054", "Column1.fields.customfield_14175",
"Column1.fields.customfield_14055", "Column1.fields.customfield_14052",
"Column1.fields.customfield_14173", "Column1.fields.customfield_14053",
"Column1.fields.customfield_14174", "Column1.fields.timeoriginalestimate",
"Column1.fields.customfield_14179", "Column1.fields.customfield_10010",
"Column1.fields.customfield_14177", "Column1.fields.customfield_14056",
"Column1.fields.customfield_14178", "Column1.fields.customfield_11742",
"Column1.fields.customfield_12949", "Column1.fields.customfield_14160",
"Column1.fields.customfield_14161", "Column1.fields.customfield_14043",
"Column1.fields.customfield_14164", "Column1.fields.customfield_14044",
"Column1.fields.customfield_14165", "Column1.fields.customfield_14163",
"Column1.fields.customfield_14168", "Column1.fields.customfield_10240",
"Column1.fields.customfield_14048", "Column1.fields.customfield_14169",
"Column1.fields.customfield_14045", "Column1.fields.customfield_14166"}),

    #"Expanded Column1.fields.customfield_13751" = Table.ExpandListColumn(#"Removed
Columns18", "Column1.fields.customfield_13751"),

    #"Expanded Column1.fields.customfield_1" = Table.ExpandRecordColumn(#"Expanded
Column1.fields.customfield_13751", "Column1.fields.customfield_13751",
{"displayName"}, {"Column1.fields.customfield_13751.displayName"}),
```

```
    #"Removed Columns19" = Table.RemoveColumns(#"Expanded
Column1.fields.customfield_1",{"Column1.fields.customfield_14046",
"Column1.fields.customfield_13750", "Column1.fields.customfield_14167"}),

    #"Expanded Column1.fields.customfield_13753" =
Table.ExpandRecordColumn(#"Removed Columns19", "Column1.fields.customfield_13753",
{"self", "value", "id"}, {"Column1.fields.customfield_13753.self",
"Column1.fields.customfield_13753.value", "Column1.fields.customfield_13753.id"}),

    #"Removed Columns20" = Table.RemoveColumns(#"Expanded
Column1.fields.customfield_13753",{"Column1.fields.customfield_13753.self",
"Column1.fields.customfield_13753.value", "Column1.fields.customfield_13753.id",
"Column1.fields.customfield_14049", "Column1.fields.customfield_13746",
"Column1.fields.environment", "Column1.fields.customfield_13748",
"Column1.fields.customfield_13747", "Column1.fields.customfield_13749",
"Column1.fields.parent", "Column1.fields.updated", "Column1.fields.description",
"Column1.fields.customfield_10011", "Column1.fields.summary",
"Column1.fields.customfield_13751.displayName", "Column1.fields.customfield_10640",
"Column1.fields.duedate", "Column1.self", "Column1.fields.fixVersions.description",
"Column1.fields.resolution.id", "Column1.fields.resolution.description",
"Column1.fields.priority.id", "Column1.fields.versions.id",
"Column1.fields.versions.archived", "Column1.fields.assignee.displayName",
"Column1.fields.status.description", "Column1.fields.status.id",
"Column1.fields.status.statusCategory.id",
"Column1.fields.status.statusCategory.colorName", "Column1.fields.creator.displayName",
"Column1.fields.reporter.displayName", "Column1.fields.aggregateprogress.progress",
"Column1.fields.aggregateprogress.total", "Column1.fields.aggregateprogress.percent",
"Column1.fields.issuetype.id", "Column1.fields.issuetype.description",
"Column1.fields.project.id", "Column1.fields.project.key",
"Column1.fields.project.projectCategory.id",
"Column1.fields.project.projectCategory.description",
"Column1.fields.project.projectCategory.name", "Column1.fields.workratio",
"Column1.fields.watches.watchCount"}),

    #"Removed Duplicates" = Table.Distinct(#"Removed Columns20", {"Column1.key"}),

    #"Renamed Columns" = Table.RenameColumns(#"Removed
Duplicates",{{"Column1.key", "Issue Key"}, {"Column1.fields.fixVersions.id", "Fix Version
ID"}, {"Column1.fields.fixVersions.name", "Fix Version Name"},
{"Column1.fields.fixVersions.archived", "Fix Version Archieved"},
{"Column1.fields.fixVersions.released", "Fix Version Released"},
{"Column1.fields.fixVersions.releaseDate", "Fix Version ReleaseDate"},
```

{"Column1.fields.resolution.name", "Resolution Name"}, {"Column1.fields.priority.name", "Priority"}, {"Column1.fields.versions.description", "Version Description"}, {"Column1.fields.versions.name", "Versions Name"}, {"Column1.fields.versions.released", "Versions Release"}, {"Column1.fields.versions.releaseDate", "Versions Release Date"}, {"Column1.fields.status.name", "Status"}, {"Column1.fields.status.statusCategory.key", "Status Key"}, {"Column1.fields.status.statusCategory.name", "Status Category Name"}, {"Column1.fields.issuetype.name", "Issue Type"}, {"Column1.fields.project.name", "Project Name"}, {"Column1.fields.customfield_11643", "Planned End Date"}, {"Column1.fields.customfield_11642", "Planned Start Date"}}),

#"Reordered Columns" = Table.ReorderColumns(#"Renamed Columns",{"Issue Key", "Fix Version ID", "Fix Version Name", "Fix Version Archieved", "Fix Version Released", "Fix Version ReleaseDate", "Resolution Name", "Priority", "Version Description", "Versions Name", "Versions Release", "Versions Release Date", "Status", "Status Key", "Status Category Name", "Issue Type", "Project Name", "Planned Start Date", "Planned End Date", "Column1.fields.resolutiondate", "Column1.fields.created"}),

#"Renamed Columns1" = Table.RenameColumns(#"Reordered Columns",{{"Column1.fields.resolutiondate", "Resolution Date"}}),

#"Removed Columns21" = Table.RemoveColumns(#"Renamed Columns1",{"Column1.fields.created"}),

#"Changed Type" = Table.TransformColumnTypes(#"Removed Columns21",{{"Fix Version ReleaseDate", type date}})

in

#"Changed Type"

FlexeLint file transformation script

let

Source = Csv.Document(Web.Contents("http://build-project.dev.mentorg.com/job/MajorTier1_project_master_Flexelint/ws/build/tmp-glibc/deploy/lint/flexelint-summary.txt"),[Delimiter=":", Columns=2, Encoding=1252]),

#"Split Column by Delimiter" = Table.SplitColumn(Source,"Column1",Splitter.SplitTextByEachDelimiter({"."}, QuoteStyle.Csv, true),{"Column1.1", "Column1.2"}),

```
    #"Changed Type" = Table.TransformColumnTypes(#"Split Column by
Delimiter",{{"Column1.1", type text}, {"Column1.2", type text}, {"Column2", Int64.Type}})

in

    #"Changed Type"
```

Flexilint SCA tranformation script

```
let

    Source = Csv.Document(Web.Contents("http://build-
project.dev.mentorg.com/job/MajorTier1_project_master_Flexelint/ws/build/tmp-
glibc/deploy/lint/flexelint-summary.txt"),[Delimiter=":", Columns=2, Encoding=1252]),

    #"Filtered Rows" = Table.SelectRows(Source, each [Column2] <> "")

in

    #"Filtered Rows"
```

Appendix B

Current MS Completion = FORMAT(([Resolved in Current Version]/[Total Issues in Current Version]),"Percent")

Current Version Name = MINX(FILTER('Major Tier1 Project',[Fix Version Released]="False" && 'Major Tier1 Project'[Fix Version ReleaseDate]=MINX(FILTER('Major Tier1 Project',[Fix Version Released]="False"),[Fix Version ReleaseDate])),[Fix Version Name])

Current Version ID = MINX(FILTER('Major Tier1 Project',[Fix Version Released]="False" && 'Major Tier1 Project'[Fix Version ReleaseDate]=MINX(FILTER('Major Tier1 Project',[Fix Version Released]="False"),[Fix Version ReleaseDate])),[Fix Version ID])

Critical in Current Version = COUNTAX(FILTER('Major Tier1 Project',[Priority]="Critical" && 'Major Tier1 Project'[Fix Version ID]=MINX(FILTER('Major Tier1 Project',[Fix Version Released]="False" && 'Major Tier1

Project'[Fix Version ReleaseDate]=MINX(FILTER('Major Tier1 Project',[Fix Version Released]="False"),[Fix Version ReleaseDate])),[Fix Version ID])),[Priority])

Critical in Next Version = COUNTAX(FILTER('Major Tier1 Project',[Priority]="Critical" && 'Major Tier1 Project'[Fix Version ID]=MINX(FILTER('Major Tier1 Project',[Fix Version Released]="False" && 'Major Tier1 Project'[Fix Version ReleaseDate]>MINX(FILTER('Major Tier1 Project',[Fix Version Released]="False"),[Fix Version ReleaseDate])),[Fix Version ID])),[Priority])

Current Version Release Date = FORMAT(MINX(FILTER('Major Tier1 Project',[Fix Version Released]="False" && 'Major Tier1 Project'[Fix Version ReleaseDate]=MINX(FILTER('Major Tier1 Project',[Fix Version Released]="False"),[Fix Version ReleaseDate])),[Fix Version ReleaseDate]), "dd.mm.yyyy")

Next MS Completion = FORMAT(([Resolved in Next Version]/([Total Issues in Next Version])),"Percent")

Resolved in Current Version = COUNTAX(FILTER('Major Tier1 Project',[Status]="Resolved" && 'Major Tier1 Project'[Fix Version ID]=[Current Version ID]),[Issue Key]) + COUNTAX(FILTER('Major Tier1 Project',[Status]="Closed" && 'Major Tier1 Project'[Fix Version ID]=MINX(FILTER('Major Tier1 Project',[Fix Version Released]="False" && 'Major Tier1 Project'[Fix Version ReleaseDate]=MINX(FILTER('Major Tier1 Project',[Fix Version Released]="False"),[Fix Version ReleaseDate])),[Fix Version ID])),[Issue Key])

Total Issues in Current Version = COUNTAX(FILTER('Major Tier1 Project',[Fix Version ID]=MINX(FILTER('Major Tier1 Project',[Fix Version Released]="False" && 'Major Tier1 Project'[Fix Version ReleaseDate]=MINX(FILTER('Major Tier1 Project',[Fix Version Released]="False"),[Fix Version ReleaseDate])),[Fix Version ID])),[Issue Key])

Next Version ID = MINX(FILTER('Major Tier1 Project',[Fix Version Released]="False" && 'Major Tier1 Project'[Fix Version ReleaseDate]>MINX(FILTER('Major Tier1 Project',[Fix Version Released]="False"),[Fix Version ReleaseDate])),[Fix Version ID])

Duration of Performed Test Cases = FORMAT(SUMX(SUMMARIZE('esd_abu_test_results majorTier1_project',[TestID],"Time Duration Average",AVERAGE('esd_abu_test_results majorTier1_project'[TestDuration])), [Time Duration Average]), "hh:mm:ss")

Total Number of Test Cases = COUNTAX(FILTER('esd_abu_testplan majorTier1_project','esd_abu_testplan majorTier1_project'[ValidStatus]="TRUE" && 'esd_abu_testplan majorTier1_project'[ValidUntil]=BLANK()),[TestID]) + 0

Failed Test Cases = COUNTAX(FILTER('esd_abu_test_results majorTier1_project',[TestResult]="FAILED"), [TestID]) + 0

Passed Test Cases = COUNTAX(FILTER('esd_abu_test_results majorTier1_project',[TestResult]="PASSED"), [TestID]) + 0

Unknown Test Cases = COUNTAX(FILTER('esd_abu_test_results majorTier1_project',[TestResult]="Unknown"), [TestID]) + 0

Accepted System Test Cases = COUNTAX(FILTER('esd_abu_testplan majorTier1_project',[VerificationStatus]="accepted" && [KindOfTest] = "System Test" && 'esd_abu_testplan majorTier1_project'[ValidStatus]="TRUE" && 'esd_abu_testplan majorTier1_project'[ValidUntil]=BLANK()),[TestID]) + 0

In Analysis System Test Cases = COUNTAX(FILTER('esd_abu_testplan majorTier1_project',[VerificationStatus]="Analysis" && [KindOfTest] = "System Test" && 'esd_abu_testplan majorTier1_project'[ValidStatus]="TRUE" && 'esd_abu_testplan majorTier1_project'[ValidUntil]=BLANK()),[TestID]) + 0

Number of Automatic Test Cases = COUNTAX(FILTER('esd_abu_testplan majorTier1_project',[VerificationMethod]="Automatic Test" && 'esd_abu_testplan majorTier1_project'[ValidStatus]="TRUE" && 'esd_abu_testplan majorTier1_project'[ValidUntil]=BLANK()),[TestID]) + 0

Test Cases with Associated Requirements = COUNTAX(FILTER('esd_abu_testplan majorTier1_project', 'esd_abu_testplan majorTier1_project'[RequirementID]<>BLANK() && 'esd_abu_testplan majorTier1_project'[ValidStatus]="TRUE" && 'esd_abu_testplan majorTier1_project'[ValidUntil]=BLANK()),[TestID]) + 0

Test Cases to Requirment Tracebility = FORMAT([Test Cases with Associated Requirements]/[Total Number of Test Cases], "Percent")

Total Number of Requirements For Tracebility = COUNTAX(FILTER('esd_abu_trs project','esd_abu_trs project'[ValidStatus]="TRUE" && 'esd_abu_trs project'[ReqState]<>"Obsolete" && 'esd_abu_trs project'[ReqState]<>"Rejected" && 'esd_abu_trs project'[ReqCategory]<>"Legal" && 'esd_abu_trs project'[ReqCategory]<>"Compliance to Standard" && 'esd_abu_trs project'[ValidUntil]=BLANK()),[RequirementID]) + 0

Requirements with Associated Test Cases = COUNTAX(FILTER('esd_abu_trs project', 'esd_abu_trs project'[TestID]<>BLANK() && 'esd_abu_trs project'[ValidStatus]="TRUE" && 'esd_abu_trs project'[ValidUntil]=BLANK()),[RequirementID]) + 0

Requirements to Test Cases Tracebility = FORMAT([Requirements with Associated Test Cases]/[Total Number of Requirements For Tracebility], "Percent")

Accepted Requirements = COUNTAX(FILTER('esd_abu_trs project', 'esd_abu_trs project'[ReqState]="Accepted" && 'esd_abu_trs project'[ValidStatus]="TRUE" && 'esd_abu_trs project'[ValidUntil]=BLANK()),[RequirementID]) + 0

In Analysis Requirements = COUNTAX(FILTER('esd_abu_trs project', 'esd_abu_trs project'[ReqState]="Analysis" && 'esd_abu_trs project'[ValidStatus]="TRUE" && 'esd_abu_trs project'[ValidUntil]=BLANK()),[RequirementID]) + 0

Issues Per 100 Lines of Code = FORMAT(VALUE(LOOKUPVALUE('Flexelin-SCA'[Column2],'Flexelin-SCA'[Column1],"Number of issues per 100 LoC ......")),"Fixed")

Total Number of Issues Found = VALUE(LOOKUPVALUE('Flexelin-SCA'[Column2],'Flexelin-SCA'[Column1],"Total number of issues found ......"))

Total Number of Lines of Code = VALUE(LOOKUPVALUE('Flexelin-SCA'[Column2],'Flexelin-SCA'[Column1],"Total number of LoC (Lines of Code)"))

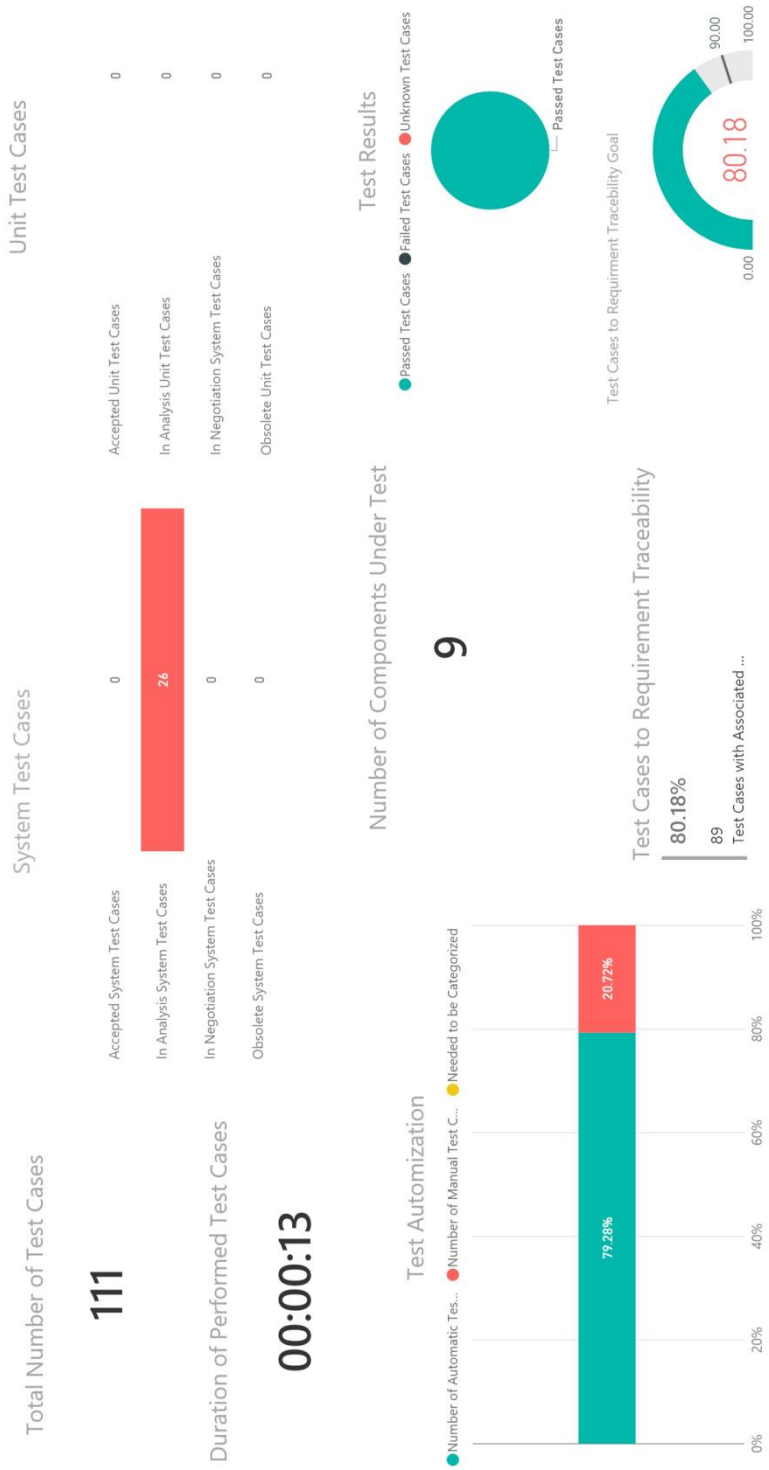
Total Number of Source Code Files = VALUE(LOOKUPVALUE('Flexelin-SCA'[Column2],'Flexelin-SCA'[Column1],"Total number of Files ....."))

Number of .c Files = COUNTAX(FILTER('Flexelint-Files-Details','Flexelint-Files-Details'[Column1.2]="c"),[Column1.2]) + 0

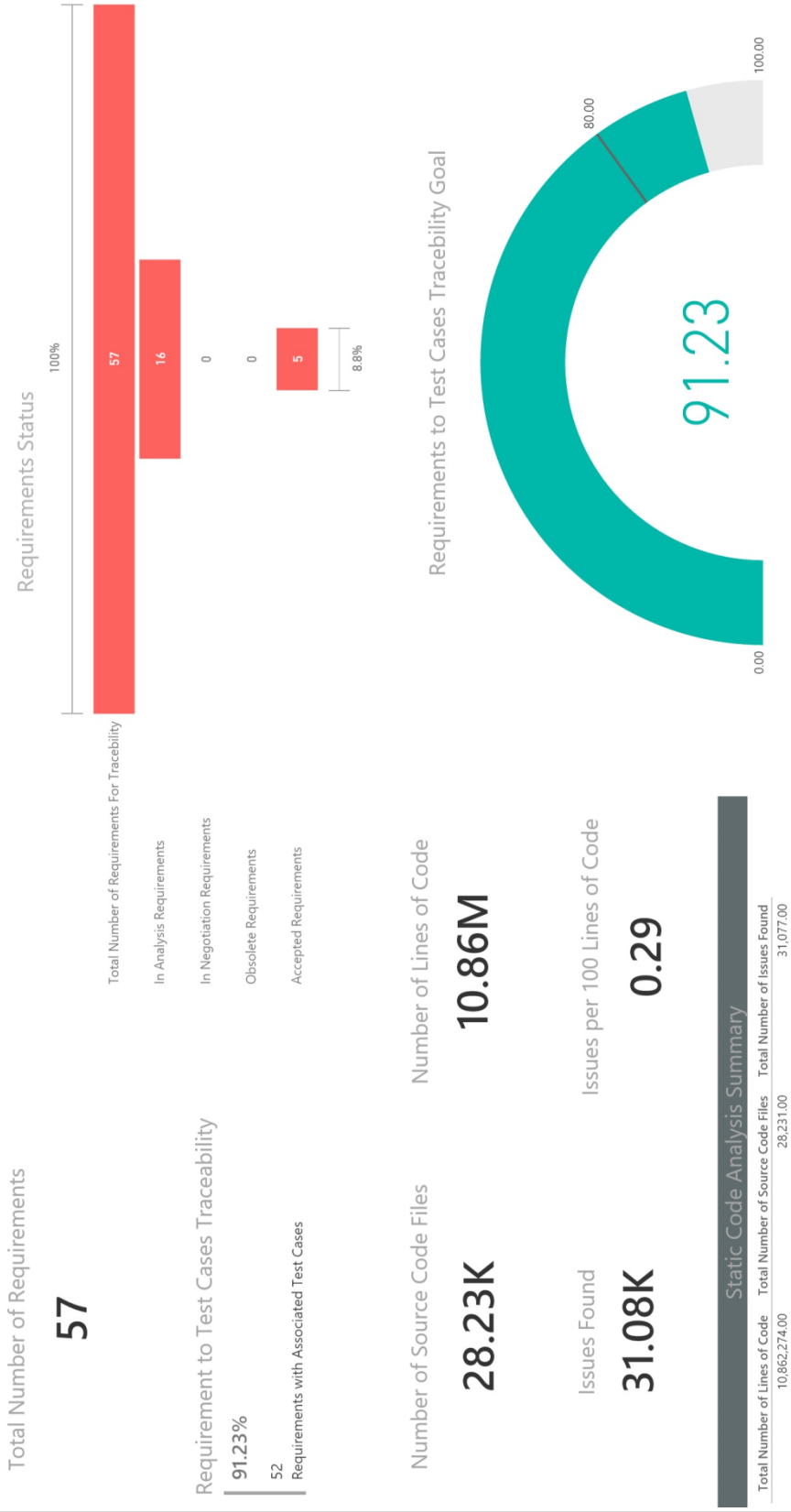
Number of header Files = COUNTAX(FILTER('Flexelint-Files-Details','Flexelint-Files-Details'[Column1.2]="h"),[Column1.2]) + 0

Appendix C

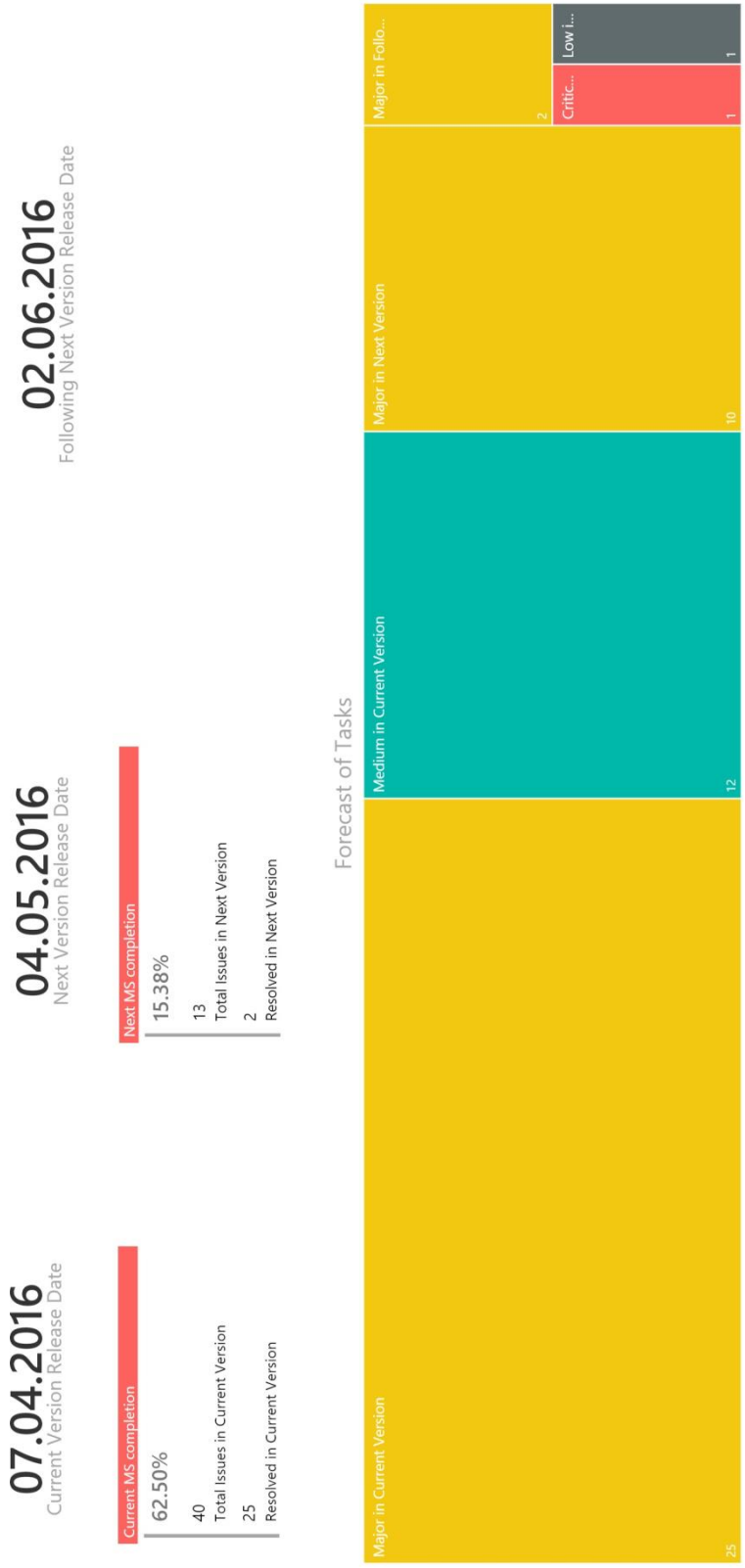
Report from testing



Report from SCA and testing



Report from JIRA



Executive dashboard